# A COMPARATIVE PERFORMANCE ANALYSIS OF WOA VS. SOA

*Ashish Verma
**Vikas Bhatnagar
***Siddharth Jain

**ABSTRACT**

Service-Oriented Architectures (SOA) is an Emerging approach that addresses the requirements of loosely coupled, standards-based, and protocol independent distributed computing. A Distributed Computing is always required a tight coupled relationship between all working services. Basically SOA provides a large number of objects that are working in modular services as reusable software components. Generally there are no any alternative for SOA to provide flexibility and reduction in the cost of services which are basically used in the IT field as reusable components. This functionality is provided by the Enterprise Service Bus (ESB) that is an integration platform that utilizes Web services standards to support a wide variety of communications patterns over multiple transport protocols and deliver value-added capabilities for SOA applications. But in this Context we are introducing the "WEB 2.0"which is used to provide reusable IT components dynamically. In this paper we will introduce the methodology of design WOA using the concept of SOA. The big picture will follow the existing SOA model. In particular, this WOA methodology comprises conceptual as well as realization issues and breaks WOA design down into three distinct phases.

**KEYWORDS:** Design Methodology, Reusable Components and Web oriented Architecture.

***Assistant Professor, Department of Computer Science & Engineering' Anand International College of Engineering, Jaipur**
****Assistant Professor, Department of Computer Science & Engineering' Anand International College of Engineering, Jaipur**
*****Assistant Professor, Department of Computer Science & Engineering' Anand International College of Engineering, Jaipur**

## INTRODUCTION

**Web-oriented architecture** (**WOA**) is a style of software architecture that extends service-oriented architecture (SOA) to web-based applications, and is sometimes considered to be a light-weight version of SOA. WOA is also aimed at maximizing the browser and server interactions by use of technologies such as REST and POX. In software engineering, a **Service-Oriented Architecture** (**SOA**) is a set of principles and methodologies for designing and developing software in the form of interoperable services. These services are well-defined business functionalities that are built as software components that can be reused for different purposes. WOA is simply a way of implementing SOA by creating services that are Restful resources, allowing any service or data to be accessed with a URI. (REST, by the way, stands for representational state transfer).

**Long version:** WOA is an architectural style that is a sub style of SOA based on the architecture of the www with the following additional constraints: globally linked, decentralized, and uniform intermediary processing of application state via self-describing messages.
**Shorthand version:** WOA = SOA + WWW + REST

**MOTIVATION**: The main remainder for this paper is to develop a Restful Application for which we have to convince the users to use WOA as a REST because rest is not a piece of software or like software packages instead of these we have following questions which are only be answered by REST which is a set of principles or rules in WOA:

- Is there any service which is mainly concentrate on resources?
- Is there any service which works on Web methods?
- Is your web service works in remote distributed environment?
- What is lifetime of URI and URL's in Web services?
- How your web service looks like in enterprises?
- Is your web service highly consumable, efficient and interoperable with support of latest widgets and gadgets and embedded social apps.

- Is there any **Simple tool to weave the Web of resources into new applications?**
- What types of objects are supported in remote basic environment?
- Is any service is called by any object if yes can we define its scope of visibility or its lifecycle?
- How is the web services protected?
- How can we define any client interface in web services?

**EXISTING SYSTEM:** Here we are presenting the algorithm in which previous applications or web services were developed before introducing the WOA (Restful Applications)

**Input**: Q. Any user Query
**Output**: Final WS*
**Step 0**: Begin
**Step 1**: Take User Input as Query (SOAP Message)
**Step 2**: Proceed this query to the UDDI (Discovery Mechanism)
**Step 3**: Now the Query will be search in the UDDI.
**Step 4**: Response if only Found and Iff Server gets registered in UDDI also.
**Step 5**: If Step 4 gets true results then describe its description by (WSDL)
**Step 6**: After Step 4 & 5 the user will get resultant WS*
**Step 7**: Then Transfer SOAP message to the Client from service provider in the form of XML.
**Step 8**: End

If we consider the above mentioned example of SOA based web service we can say that the main concentration of this service is on Functional Data it does not matter from where data is coming or from where that particular service is being analyzed this only consider the functional the functional part of the service.

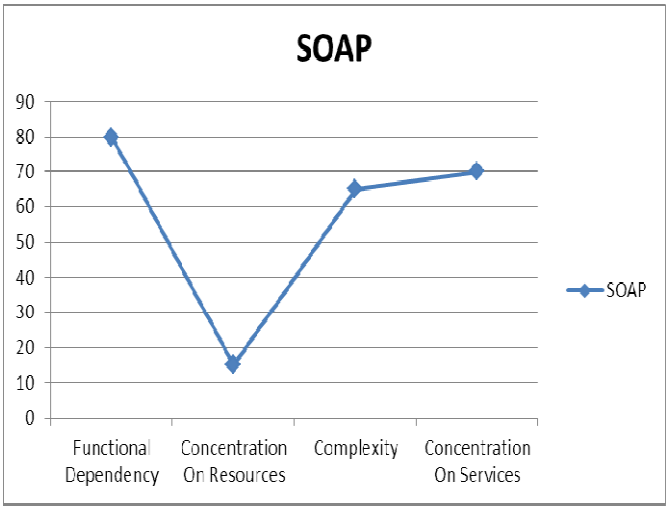


**Figure 1.1: Resulting graph after applying previous algorithm**

## PROPOSED SYSTEM

**The proposed system is known as RESH Algorithm.**

From the previous algorithm we are ready to develop SOA application in the SOAP environment and now we are giving a proposed algorithm for developing any web services in WOA architecture with the use of REST architecture style:

***RESH Algorithm for Developing RESTFul Application***

**Input**: Q. Any Query from Client.

**Output**: Resultant WS*

**Step 0**: Begin

**Step 1**: Identify all the Resources by URI's

**Step 2**: Convert Verbs in to the Noun.

**Step 3**: After applying Step-2 get all Correct URI's

**Step 4**: Categorize them and apply suitable web method (GET, PUT, POST, and DELETE)

**Step 5**: Exposing them in URI Directory.

**Step 6**: Transfer them in XML Message Format.

**Step 7**: Repeat Step 2, 4 then and Perform Step 5 and 6.

**Step 8**: End

**AN EXAMPLE BASED ON RESH ALGORITHM**

**Step 1**: GET http://adduser?name=”Ashish”

After getting this URI we can see that it is not proper in terms of RESTful application so we need to convert it just need to convert all verbs in to the noun.

**Step 2:** GET http://user/Ashish

**Step 3**: GET/user/Ashish

host: myserver

type:application/XML

**Step 4:** Now we need to apply web methods because the main purpose of this URI is to get data and ADD it in to the Database so we need to apply POST web method.

POST /User/HTTP 1.1

host :myserver

<?xml version=”1.0” ?>

<user>

<name>Ashish</name>

</user>

**Step 5**: Expose it if we have any other URI.

**Step 6**: Transfer them in to the XML format as per client Interface.

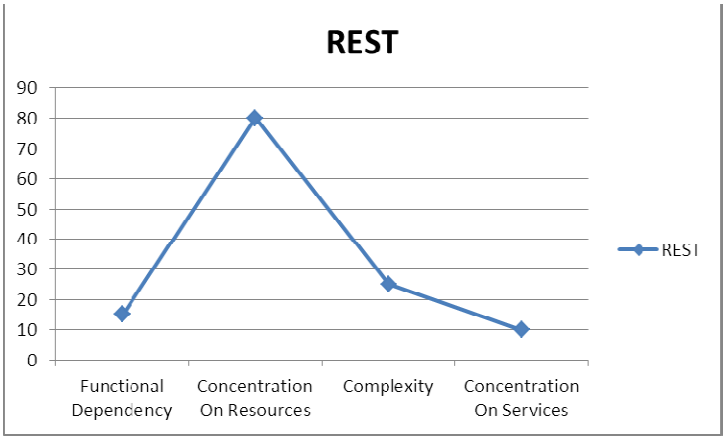


**Figure 1.2: Resulting graph after applying RESH algorithm**

**THE COMPARATIVE ANALYSIS BETWEEN SOA AND WOA:**

While comparing SOA and WOA we are taking some parameters one of them is round trip latency in a network, latency, a synonym for delay, is an expression of how much time it takes for a packet of data to get from one designated point to another as much as latency increases it directly impacts on the Quality of Service (QoS) and another parameter for comparing is the packet size of each Request / Response pair. So when we conducted these tests (Add, Update, get and Remove) comparison we found that total round trip time latency and average packet size was low in the case of Rest than the Traditional RPC based service or we can say service oriented Architecture. And it is clear that SOAP messages, across the board, were larger than their REST equivalents.

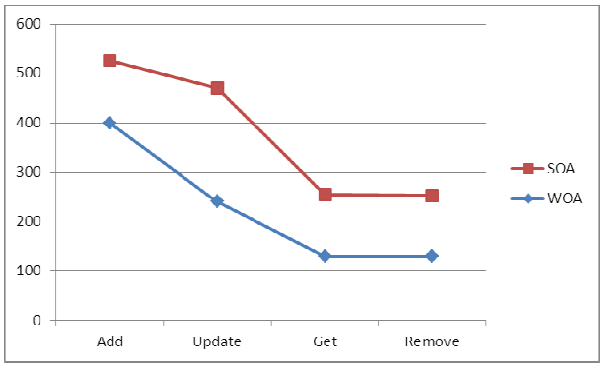


Fig 1.3:**Benchmark Test/Add application Profile Service (Latency Graph)**

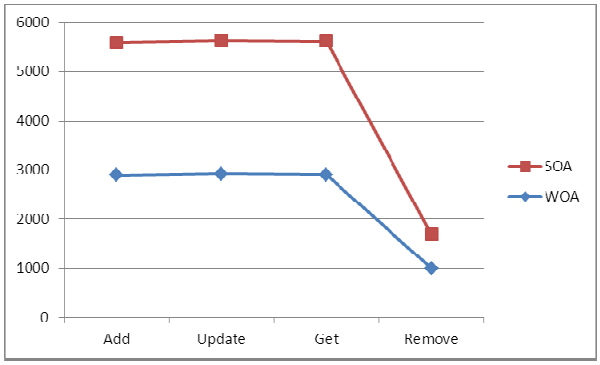


Fig 1.4: **Benchmark Test/Add application Profile Service (Packet Size Graph)**

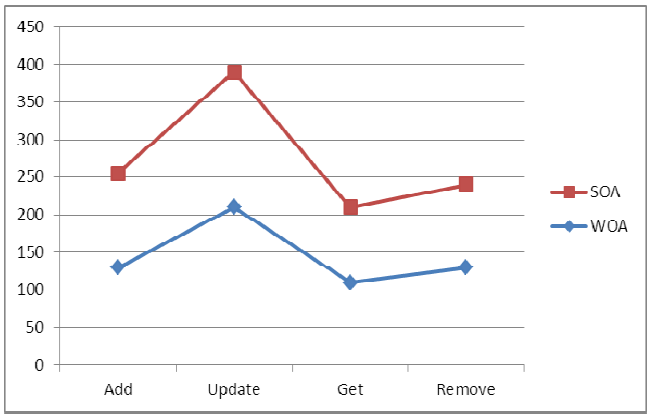


Fig 1.5:**Benchmark Test/Device Profile Service (latency Graph)**

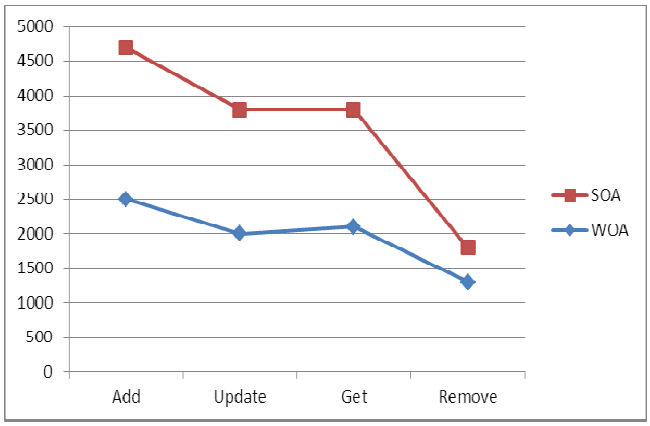


Fig 1.6: **Benchmark Test/Device Profile Service (Packet Size Graph)**

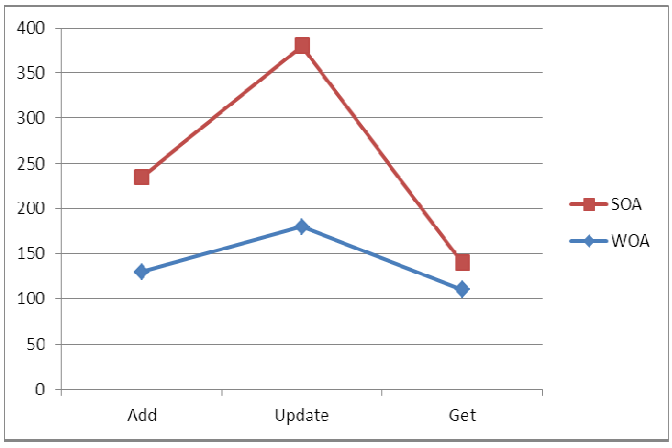


Fig 1.7: **Benchmark Test/User Account (Latency Graph)**

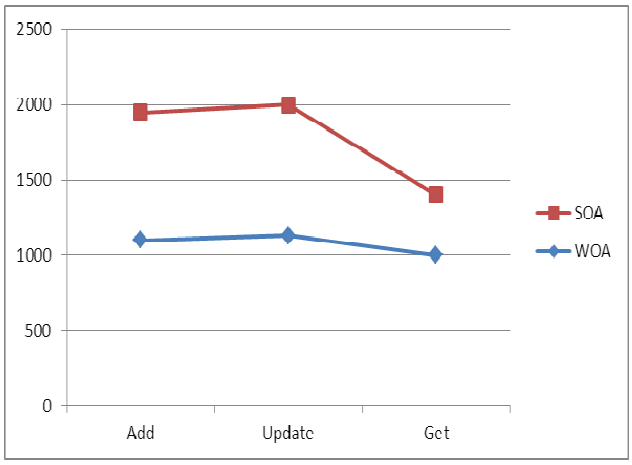


**Fig 1.8: Benchmark Test/User Account (Packet Size Graph)**

REST request and response messages add zero overhead to the messages being transmitted apart from the standard HTML headers which are used to route the packets through the network. SOAP, on the other hand, encloses each message payload within an additional SOAP 'envelope' set of XML tags and adds a few SOAP-related headers to the outbound HTTP packet. REST is able to take advantage of simplistic CRUD situations and execute them much more efficiently than the SOAP implementation. Some examples of this are the 'Remove' services for application profiles, device profiles, and user accounts. For these services, all that's required to delete an item from the back-end data model is the item's ID number. Therefore, the REST implementation merely sends an HTTP DELETE command to the appropriate resource URL with the ID number prepended. By doing so, it doesn't have to include any internal XML payload to represent this ID number and, thus, cuts down on its overall packet size.

**SOA VERSUS WOA**

- SOAP always transfer message in the form of its normal structure.
- On the other hand REST always transfers its message in the form of URI.
- Same from the response SOAP will always add some additional information on the payload of messages.
- But on the other hand REST simply transfers its message in the form of URI's.

**REFERENCES**

[1].THE SOA WITH REACH: WEB-ORIENTED ARCHITECTURE BY DION HINCHCLIFFE.

[2]. DESIGN AND DEVELOPMENT OF A REST-BASED WEB SERVICE PLATFORM FOR APPLICATIONS INTEGRATION BY LUIS OLIVA FELIPE

[3]. A GUIDE TO DESIGNING AND BUILDING RESTFUL WEB SERVICES WITH WCF 3.5 AARON SKONNARD, PLURALSIGHT OCTOBER 2008

[4]. WEB SERVICES, PART 1: SOAP VS. REST BY BRENNAN SPIES

[5]. WEB SERVICES, PART 1: SOAP VS. REST BY BRENNAN SPIES, OLAF ZIMMERMANN, FRANK LEYMANN

*[6]. RESTful Web Services: A Reviewby Kurt Cagle in Reviews*

[7]. SURVEY: SOA DELIVERING; SOAP OUT, REST IN BY JOE MCKENDRICK FOR SERVICE ORIENTED

[8]. REST BATTLES SOAP FOR THE FUTURE OF INFORMATION SERVICES BY JOHN NEWTON

[9].THE FUTURE OF RESTFUL DRUPAL BY WSCCI TEAM

[10].RESTFUL WEB SERVICES BY LEONARD RICHARDSON AND SAM RUBY

[11]. REST AND WEB SERVICES: IN THEORY AND IN PRACTICE BY PAUL ADAMCZYK, PATRICK H. SMITH, RALPH E. JOHNSON, AND MUNAWAR HAFIZ

[12]. REST-AN INTRODUCTION BY DANIEL SILVA

[13]. THE DESIGN OF A RESTFUL WEB-SERVICE BY NADIA MOHEDANO TROYANO

[14]. DESIGNING RESTFUL WEB APPLICATIONS BY BEN RAMSEY.