Performance comparison of SOAP and REST based Web Services for Enterprise Application Integration

Smita Kumari
Department of Computer science and Engineering
National Institute of Technology
Rourkela, INDIA 769008
Email: smitu2410@gmail.com

Santanu Kumar Rath
Department of Computer science and Engineering
National Institute of Technology
Rourkela, INDIA 769008
Email: skrath@nitrkl.ac.in

***Abstract*—Web Services are common means to exchange data and information over the network. Web Services make themselves available over the internet, where technology and platform are independent. Once web services are built it is accessed via uniform resource locator (URL) and their functionalities can be utilized in the application domain. Web services are self-contained, modular, distributed and dynamic in nature. These web services are described and then published in Service Registry e.g., UDDI and then they are invoked over the Internet. Web Services are basic Building blocks of Services Oriented Architecture (SOA). These web services can be developed based on two interaction styles such as Simple Object Access Protocol (SOAP) and Representational State Transfer Protocol (REST). It is important to select appropriate interaction styles i.e., either SOAP or REST for building Web Sevices. Choosing service interaction style is an important architectural decision for designers and developers, as it influences the underlying requirements for implementing web service solutions. In this study, the performance of application of web services for Enterprise Application Integration (EAI) based on SOAP and REST is compared. Since web services operate over network throughput and response time are considered as a metrics parameter for evaluation.**

Key Words- SOA, Web Service, EAI, SOAP, REST.

I. Introduction

Service-oriented architecture (SOA) is a technology neutral, platform independent design, which provides the aspects of reusability, agility, loose coupling and interoperability with the help of a collection of services. The main advantage of building services is that they provide a common way of interaction. Services can interact regardless of the different platform. Services publish its functionalities through WSDL in case of SOAP and URI in case of REST. But keep their implementation details private [1]. The property of Web services is particularly applicable to a distributed heterogeneous environment, based on certain interaction patterns like SOAP and REST, The interactions between services can involve either information passing or common data sharing or it could even involve the integration of two or more services to accomplish a particular functionality. To integrate web services, Enterprise Service Bus (ESB) has been used for integrating these services. Business Process Execution Language (BPEL) is used for the composition of SOAP-based services. Web Services consist of three main components such as service provider, service consumer and service registry. This Architecture is shown in figure 1. In SOAP based service, web services have a unique web service description language (WSDL). Which helps to

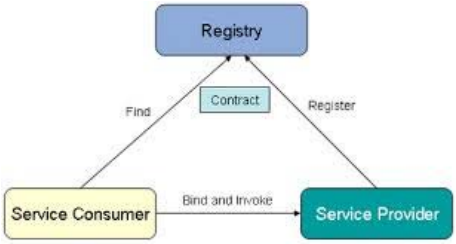

Fig. 1: Service Oriented Architecture

document the contract between service provider and service consumer. However in REST, each service is represented in terms of resource. Each resource has a unique resource identifier (URI) through which services are accessed. It is observed that for Enterprise Application integration, REST outperform SOAP, But building application in REST paradigm is difficult due to the lack of adequate tool support. While the major software infrastructure providers such as Microsoft and IBM provide tool support for developing SOAP-based Web services [2]. For instance, various platforms offer standard libraries to create SOAP services or SOAP clients, such as Axis2/Java [3], But the parsing of SOAP messages can be memory and computation intensive. REST uses client-server architecture. REST is not restricted to any particular protocol. It can use SMTP, FTP, HTTP etc. but most commonly used protocol is HTTP. In this case study, HTTP protocol has been considered for REST. REST can have message format in XML as well as JSON. On the other hand, SOAP relies on XML only. XML is verbose in nature so it increases the load on the server and increases network traffic. On the contrary, REST is lightweight in nature. In this paper, a case study of Loan Broker Application has been considered. This application is implemented in both interaction style SOAP and REST. Concurrent users are simulated with the help of Multithreading concept. Where each user is equivalent to one thread.

II. Literature review

Cesare Pautasso et al. made a technical comparison between SOAP and REST, and prescribed included architectural principles, conceptual as well as different technology decisions [4]. Another work is done by Michael Zur Muehlen et al. provide a comparison that focuses on the choreography standards of the two interaction styles [5]. They have considered

978-1-4799-8792-4/15/$31.00 ©2015 IEEE

the standard process and specification. However, it does not provide practical information on the differences between the two interfaces in terms of concrete application development. Gmez and Miguel have addressed the general issue of SOAP-based web service multimedia conferencing in the IP Multimedia Subsystem [6]. IMS is the service delivery platform of next-generation telecommunications networks. They have used SOAP to develop web interface. In which a client in a session can act as an endpoint. The client can receive, parse and send SOAP messages. David Lozano et al. have developed multimedia conferencing applications using RESTful Web service [7].

In one of the research work by Tekli et al. have provided methods to enhance SOAP performance. The basic idea is to identify common parts of SOAP messages so that these common parts are processed only once hence, avoiding a large amount of overhead [8]. SOAP processing include message parsing, serialization, deserialization, compression, multicasting and security evaluation. Every phase is discussed in detail to enhance SOAP performance.

In 2012, Potti et al. have provided a comparative performance analysis in term of throughput and response time between SOAP and REST. They have designed SOAP and REST-based web services for performing Create, Read, Update, and Delete (CRUD) operations on a database and retrieving local files. Metrics such as response time and throughput have been used to compare the performance of these Web services [2].

In 2013, comparison on the performance of SOAP and RESTful web services based on different metric for the mobile environment and a multimedia conference was taken into consideration by Snehal and Puja Padiya [9]. It is observed that the article available in the literature provide an opportunity that can be explored [2]. In their study, SOAP and REST for CRUD operation is designed, and performance is compared. But the most result was inconclusive. The reason behind this the smaller sample has been used in the experiment. Payload (customer data and image files) used for CRUD operations may not have been sufficient to create substantial differences between REST and SOAP interaction styles.

In this study, Enterprise application integration has been done in the both SOAP and REST styles. A case study on LoanBroker Application is used. In this work, the sample size is comparatively large, and the result provides a substantial difference between SOAP and REST.

III. Overview of OpenESB And LoanBroker Application

A. Tool for building SOA application

OpenESB is the easiest and the most efficient ESB tool for building and i ntegration of SOA applications [3]. Enterprise Service Bus (ESB) is a larger integration concept involving intelligent routing, XML transformation, federated management, orchestration and process flow [3]. ESB is consistent with SOA principles. The logic of using this bus architecture is to link each service directly to the bus. Thus, large numbers of point-to-point connections between applications and services are avoided. This conforms with the agility that is the main principle of SOA [10].

Concurrent service requests are handled with the help of multi-threading. Total of 28 number of simultaneous users are simulated. For each user throughput and response time is calculated then the average of response time and throughput is taken as a final result.

IV. Case Study

In this section, standard example of LoanBroker is discussed [11]. LoanBroker example captures the various aspect of service compositions. In this example, consumer makes a request for best loan quote among the different banks. LoanBroker example clearly demonstrates a full ESB architecture.

A. Message Flow in LoanBroker System

Specification of requirement [11].

- Customer sends a request to LoanBroker application system with the required loan amount.
- Loan Broker system sends the request to CreditAgency service to get the credit profile of the customer.
- CreditAgency service sends the credit profile of the customer to LoanBroker system.
- LoanBroker system send the credit profile as input to the LenderGateway services.
- LenderGateway services send the lists of the bank to the LoanBroker system based on the credit profile of the customer and the amount of loan requested.
- LoanBroker system sends the request to all the banks.
- Bank services compute the loan rate and return to the LoanBroker.
- LoanBroker system sends loan rate to the getBestLoanQuote service. which return the best loan rate.
- Appropriate loan rate is returned to the customer after getting the best loan rate.

B. Type of Services Used in building LoanBroker system

LoanBroker application consists of two types of services.

1. Primary services
2. Composite Services

Primary services are the atomic services. These type of services can not be further partitioned. In the loan Broker application, most of the services are atomic except LoanBroker service that is composed of other services. Table I shows the list of various services involved in it.

Loan Broker application is the result of the integration of different primary services. In SOAP interface, integration of these services is achieved using BPEL engine of OpenESB tool. In case of REST, services are integrated by taking the output of one service as input to the other with the help of JSP. In SOAP, for service integration direct support of BPEL module is available, and it composes the services with the assistance of WSDL address. REST is resource oriented platform in which Web application Description Language (WADL) is generated, and it can not used in BPEL Engine. BPEL only supports WSDL address.

TABLE I: Functionalities of services with requisite input and output

Services	Input	Output
$CreditAgency$	LoanAmount	CreditProfile
$LenderGateway$	CreditProfile	List of Banks
$Bank1$	CreditProfile and LoanAmount	Return Rate
$Bank2$	CreditProfile and LoanAmount	Return Rate
$Bank3$	CreditProfile and LoanAmount	Return Rate
$BestLoanQuote$	rates return from all banks	Return best Rate
$LoanBroker$	LoanAmount	Return best Rate

V. Proposed Work

To select appropriate interaction style for building web services. The web services are accessed over the network. Metrics such as throughput and response time has been considered as performance parameter for evaluation. In this research work, performance evaluation for enterprise application is provided. However, the performance of these two techniques are provided for Create, Read, Update, Delete(CRUD) operation in one of the work by [2]. In this paper, a comparison between SOAP and REST for enterprise application integration (EAI) is provided. A case study on LoanBroker application has been considered.

A. SOAP Based Web Services

For developing the SOAP based services, open ESB tool has been used. All the services are coded in Java programming language. MySQL database is used. This database contains the files such as credit profile, of the customer available balance and customer id as a primary key. The customer with a unique customer id and loan amount sends the request to the LoanBroker application and gets best loan quote as a response. LoanBroker application results from the integration of all other services. This application has client-server architecture. In this study, we have considered that the single machine acts as both client and server. So the network latency can be considered as zero. After the services are deployed and run successfully, it produces a WSDL address. This WSDL address is SOAP endpoint, through which SOAP messages are exchanged between the services.

B. Integration of SOAP Based Services using BPEL Engine

The Loan Broker application is considered as example of SOA/ESB [11].

It is discussed that how SOAP and REST behave for integration of large application. LoanBroker application is integrated using the BPEL process. A process that is specified in BPEL consists of two types of activities such as basic activities and structured activities [1]. Basic activities cover receive, reply, wait, switch, while, and sequence. The structured activities determine the structure, of the sequencing of the process, and the basic activities determine what happens in the process [13]. BPEL process takes the WSDL address and based on these addresses these services are accessed in the order in which they are integrated. The BPEL module is shown in figure 2 depicts the integration of all the services. This application receives a series of input and produces desired output. Composite application is made from the BPEL Engine using CASA (Composite Application Service Assembly). This

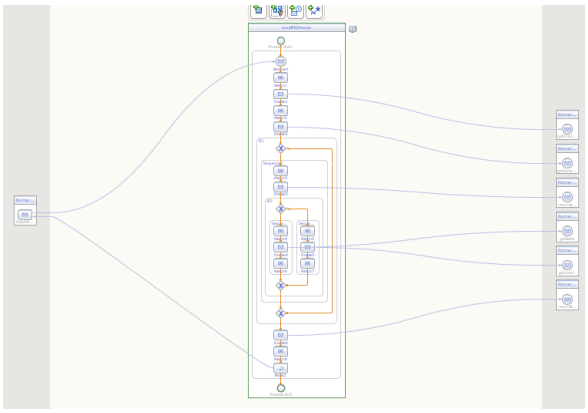

Fig. 2: BPEL module of LoanBroker application

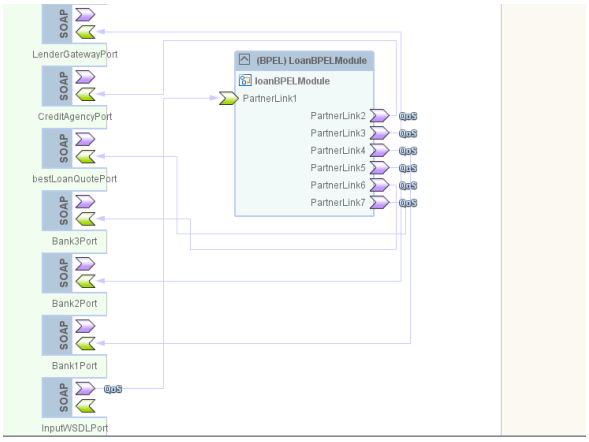

Fig. 3: Composite application Service Assembly

contains SOAP messages and input-output port as shown in figure 3. This is automatically generated from the BPEL module.

C. REST based web services.

Four important system properties of RESTful services are being highlighted as:
(1) Uniform interface
(2) Addressability
(3) Statelessness
(4) Connectedness
In RESTful Web services, these properties are embodied in resources and URIs representations [14]. REST based web services are developed with the help of OpenESB tool. It uses the same database that is used by the SOAP. Each service is represented in term of the resource. Each service possess unique Uniform Resource Indicator (URI).

D. Integration of REST based Services

REST based services, being identified using its URI does not generate WSDL address. It generates WADL address. But this WADL address cannot be used with BPEL engine, as BPEL only supports WSDL address. BPEL engine present in

OpenESB tool doesn't provide any means for integration of REST based services. So in this study, services are integrated withthe help of JSP. The input of one service is passed as output to the other service. This is achieved through JSP. However, there are tools available to integrate services based on REST style such as MULE ESB. But the result obtained using this tool on REST can not be compared with the result obtained by OpenESB for SOAP style. So OpenESB tool is used in the development of both types of services for proper comparison.

VI. Performance Metrics.

For measuring throughput and response time files of different size ranging from 1000 kb to 7000kb is taken. Total of 28 concurrent clients are simulated. For each request sent to web service and the change in the response time and throughput is evaluated. In the other analysis response time and throughput are analysed with the help of the number of the service request. As the number of requests go on increasing what will be the effect on throughput and response time.

A. Response Time

Response Time is the time elapsed between the request and reply. When the client application sends a request to service with appropriate input, the server takes input to process the data and passes through a series of services because LoanBroker application is integrated using all these services. After the request is processed, the resulted output is sent as a reply to the client. The simultaneous request is sent to the server using 28 clients at a time. Response Time is calculated for different file size and different number of the service requests.

$$R_t = t_1 - t_2 \tag{1}$$

where R_t = Response Time
t_1 = Time at which client sent request.
t_2= Time at which client receives response.

B. Throughput

Throughput is defined either in terms of number of the requests or in terms of average data bytes per second. Throughput is defined as a number of request per second. In this study, we have considered twenty-eight number of requests. Throughput is calculated using the total no of requests divided by total time taken to process all request.

Throughput = Number of simultaneous request send by client/Total time taken to process all the requests.

It can also defined as the total no of bytes sent per second. Suppose in processing a request it needs X bytes of data to be transferred. It takes the total of 'Y' time in transferring 'X' bytes of data. Then throughput can be calculated as.

$$T = X/Y \tag{2}$$

Where T = Throughput
X = Data Byte transferred.
Y= Time Time taken to transfer data byte.
In this files of different size ranging from 1000kb to 7000kb has been stored on disk. These files are retrieved from the disk and total time elapsed is calculated.
Throughput is calculated by dividing the file of different size by the total time elapsed. After finding the value of Response time and Throughput of SOAP and REST based web services, a comparison is made between these two techniques to have a critical assessment of the performance of both the styles.

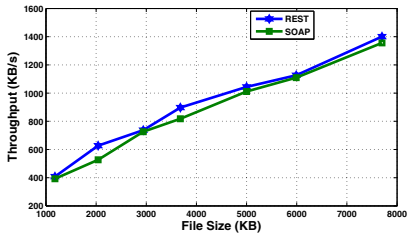

Fig. 4: Throughput comparison for different file size

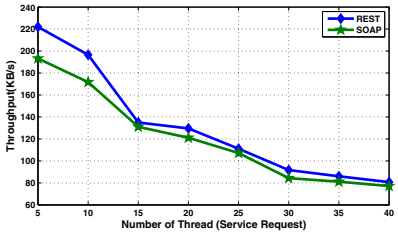

Fig. 5: Throughput comparison for different service request

VII. Comparison and Result Analysis

The proposed work in this paper compares SOAP and REST styles for Enterprise Application Integration. In this research, seven number of services namely CreditAgency, LenderGateway, Bank1, Bank2, Bank3, getBestLoanquote, and LoanBroker service (Composite service) are implemented. Services are compared using throughput and response time for the integrated application. The main idea is to implement them in SOAP and REST paradigm separately and then compare their throughput and response time.

A. Throughput comparison of SOAP and REST for EAI

The figure shown in figure 4 provides the throughput comparison of SOAP and REST for composite services. i.e, for Loan broker Application.

From the figure 4 it is evident that REST has better throughput than SOAP. As the file size increases SOAP throughput is less as compared to REST. This is because SOAP is heavyweight as its XML presentation is verbose. On the other hand, REST is lightweight. As the file size increases throughput increases for both SOAP and REST but REST has better throughput than SOAP.

In figure 5, the comparison between SOAP and REST is provided for a different number of request and as the number of request increases SOAP throughput decreases with more pace than REST.

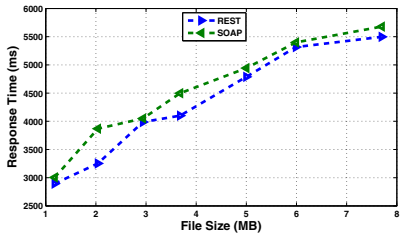

Fig. 6: Response time comparison for different file size

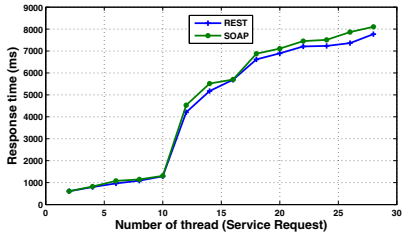

Fig. 7: Response time comparison for different number of service request

B. *ResponseTime comparison of SOAP and REST for EAI*

The figure shown in figure 6 and figure 7 provides the Response time comparison of SOAP and REST for composite service. i.e, for Loanbroker Application. Response time is measured in ms (milliseconds). For the response time, as the number of clients are increased in SOAP, response time increases but increase in number of clients doesnot affect response time of REST services drastically. REST has better response time than SOAP.

In the figure 6, comparative analysis for response time is provided. Response time is calculated for file size ranging from 1000kb to 7000kb. Response time is calculated in milliseconds.

For SOAP, response time is initially greater as compared to REST but as the file size increases for the file of size 3 MB their response time are almost equal. For other cases, SOAP has greater response time than REST.

In figure 7, as the number of service request increases SOAP response time decreases more rapidly than REST. It means that REST can handle the large number of request in the more efficient way than SOAP.

VIII. Conclusion

This paper provides a comparative analysis of the SOAP and REST styles. For the implementation of web services, it is crucial to choose an appropriate technology to build web services as it affects the underlying implementation and the integration of Enterprise application.

Integration of heterogeneous application across the enterprise is of prime importance.

In this study, LoanBroker example has been considered as a case study. LoanBroker example is suitable for evaluating ESB architecture, messaging flow and various components involved.

From the analysis, it is observed that REST based development has better throughput and response time that of the system developed based on SOAP. However, it is convenient to build the application with SOAP due the adequate support of tools. But in case of REST right kind of tools are not available. REST is an emerging technology whereas SOAP is already popular and used in building a good number of applications. Hence, it may be opined that use of REST style has the definite advantage over the use of SOAP style.

References

[1] J. Gortmaker, M. Janssen, and R. W. Wagenaar, "The advantages of web service orchestration in perspective," in *Proceedings of the 6th international conference on Electronic commerce*, pp. 506–515, ACM, 2004.

[2] P. K. Potti, S. Ahuja, K. Umapathy, and Z. Prodanoff, "Comparing performance of web service interaction styles: Soap vs. rest," in *Proceedings of the Conference on Information Systems Applied Research ISSN*, vol. 2167, p. 1508, 2012.

[3] "Open enterprise service bus." http://www.open-esb.net, August 2014.

[4] C. Pautasso and E. Wilde, "Why is the web loosely coupled?: a multi-faceted metric for service design," in *Proceedings of the 18th international conference on World wide web*, pp. 911–920, ACM, 2009.

[5] M. Zur Muehlen, J. V. Nickerson, and K. D. Swenson, "Developing web services choreography standardsthe case of rest vs. soap," *Decision Support Systems*, vol. 40, no. 1, pp. 9–29, 2005.

[6] M. Gómez and T. P. de Miguel, "Advanced ims multipoint conference management using web services," *Communications Magazine, IEEE*, vol. 45, no. 7, pp. 51–57, 2007.

[7] D. Lozano, L. A. Galindo, and L. García, "Wims 2.0: Converging ims and web 2.0. designing rest apis for the exposure of session-based ims capabilities," in *Next Generation Mobile Applications, Services and Technologies, 2008. NGMAST'08. The Second International Conference on*, pp. 18–24, IEEE, 2008.

[8] J. M. Tekli, E. Damiani, R. Chbeir, and G. Gianini, "Soap processing performance and enhancement," *Services Computing, IEEE Transactions on*, vol. 5, no. 3, pp. 387–403, 2012.

[9] S. Mumbaikar, P. Padiya, *et al.*, "Web services based on soap and rest principles," *International Journal of Scientific and Research Publications*, vol. 3, no. 5, 2013.

[10] C. Groba, I. Braun, T. Springer, and M. Wollschlaeger, "A service-oriented approach for increasing flexibility in manufacturing," in *Factory Communication Systems, 2008. WFCS 2008. IEEE International Workshop on*, pp. 415–422, IEEE, 2008.

[11] B. W. Gregor Hohpe, *Enterprise Integration Patterns*. Addison Wesley, 2003.

[12] P. Brebner, "Service-oriented performance modeling the mule enterprise service bus (esb) loan broker application," in *Software Engineering and Advanced Applications, 2009. SEAA'09. 35th Euromicro Conference on*, pp. 404–411, IEEE, 2009.

[13] A. Karande, M. Karande, and B. Meshram, "Choreography and orchestration using business process execution language for soa with web services," *International Journal of Computer Science Issues IJCSI*, vol. 11, pp. 224–232, 2011.

[14] L. Richardson and S. Ruby, *RESTful web services*. 2008.

[15] F. Brito e Abreu and W. Melo, "Evaluating the impact of object-oriented design on software quality," in *Software Metrics Symposium, 1996., Proceedings of the 3rd International*, pp. 90–99, IEEE, 1996.

[16] C. J. Burgess and M. Lefley, "Can genetic programming improve software effort estimation? a comparative evaluation," *Information and Software Technology*, vol. 43, no. 14, pp. 863–873, 2001.

[17] M. Arroqui, C. Mateos, C. Machado, and A. Zunino, "Restful web services improve the efficiency of data transfer of a whole-farm simulator accessed by android smartphones," *Computers and Electronics in Agriculture*, vol. 87, pp. 14–18, 2012.

[18] M. Halstead, *Elements of Software Science*. New York, USA: Elsevier Science, 1977.