ENHANCING THE WEB SERVICE AVAILABILITY USING SERVICE REPLICATION OVER MULTIPLE PROTOCOLS

Sundharam R.[1], **Lakshmi M.**[2]

[1]Research Scholar, Sathyabama University
Prof. & Head, Dept. of CSE, Sathyabama University
Email: [1]sun_moonr@yahoo.com , [2]hodcse@sathyabamauniv.ac.in

ABSTRACT

In the distributed computing systems such as web services, a slow service is merely equivalent to an unavailable service. So it demands for effectively increasing the availability of web service. In order to increase the availability of web service for mission critical applications we proposed an architecture [1]. In this paper we have implemented this architecture with replication of services on different protocols (HTTP, JMS and SOAP). Replication is a way by which availability of web service is increased for wide area network. Multiple requests sent by the clients to the service gateway and it multicast these request to the service replicas and waiting for the response from one or all of the services. The service gateway collects the response and analyzed for faults and a faultless response is returned back to the client thereby the availability of service is increased.

Keywords: Web service, High-availability, Replication

I. INTRODUCTION

Web services are emerging and very popular technology for providing and combining functionalities in distributed systems. A web service is a URI based accessible application which offers capabilities such as publications, discovery, selection and binding. Many service providers have adopted standards like Web Service Description Language (WSDL), Simple Object Orient Protocol (SOAP) and Universal Description Discovery and Integration (UDDI) [5] to develop and offer whole range of services to the industries such as financial, telecommunication, media and entertainment. Now-a-days web service earned more acceptability and popularity. Many of the organization begin to deploy mission critical applications using web services. Mission Critical Systems (MCS) are time specific and more deterministic and predictable. Every task has a critical time and must be completed within that time. If this application fails for any length of time that leads to great business loss. For example, Stock Exchange Services, Financial and Banking Services .The Web Services for these Systems should remain available on 24/7 basis. The availability of these systems (MCS) must be guaranteed in case of failures and network disconnections. So far little bit of work is done on mission critical application but not much work has been reported on how to ensure that a web service is available for mission critical applications. In paper [2] we proposed architecture in order to enhancing the availability of web service with the concept of web service replication. Here service replication is a way by which availability of web service is increased over a wide area network. An architecture presented in paper [2] also applied for non mission critical applications but it is too costly to implement it in practical.

This paper is structured as follows: Section 2 first we provide an overview of high availability and next we describe related past and ongoing work in this area also WS-Replication is described. In section 3 we describes about enterprise service gateway architecture to enhance the high availability of web services. We presented an evaluation and performance in section 4. Finally conclusion and future work are presented in Section 5.

II. BACKGROUND OVERVIEW

2.1 High Availability

A service that is frequently unavailable may have negative effects on the reputation of the service provider that leads to loss of business opportunities. From the user's point of view, a service that gives poor quality is virtually equivalent to an unavailable service. High availability is not an easily quantifiable term. It is system's ability to perform its function continuously without interruption for significantly longer period of time. Generally the term availability is " the proportion of time a system is in a functioning condition"

(Wikipedia online Dictionary). A simple equation to calculate availability[6]:

$$A = MTBF / (MTBF + MTTR) \quad (1)$$

Where A is the degree of availability expressed as a percentage, MTBF is the meaning between failures and MTTR is the maximum time to repair or resolve a particular problem. Some simple observations: As MTTR approaches zero, A increased towards 100 percentage. As MTBF gets larger, MTTR has less impact on A.

It is not simply limited to web services but it needs to examining infrastructure availability, middleware availability and application availability, for example failure can occur between servers, networks and disk arrays. Generally availability is specified in nines notation, for example 3-nines availability corresponds to 99.9%. In this paper we focused on services availability using middleware by introducing replication of services and have enterprise gateway. This gateway controls the flow of client request and response to and from the replicated services. Mission critical systems meets five 9's criteria i.e., system is up 99.999% of the time.

2.2 Related Work

The paper [1] introduces an enterprise level gateway. The gateway is constantly monitors the health of the services and its responsibility is to forward a client request to an appropriate service. In case that particular service is not available the gateway decides to send the request to another service. The gateway can keep check the availability of services in number of ways. However it becomes a performance overhead for the gateway. Since the gateway is responsible to monitor the health of the services it may impact the performance and the throughput. An architecture is presented in paper [2], which overcomes this and where there is replica of services available to the gateway. It simply needs to multicast the request to the available services and response is collected from all the services and the first faultless response is sent back to the requester. This architecture is implemented, tested also its performance is analyzed in paper [3] which provides success rate of 99.99% of service availability.

A frame work for highly available web service, WS-Replication has been presented in paper [4]. Jorge Salas et al. discuss a replication component which is used to replicate the web services. It's capable of enabling the stateful services with persistent state. WS-Replication allows deployment of replicated web service through web service technology and SOAP for transporting information. We are using the web service replication in different flavors. We need a different kind of replication services, because our services are deployed on different protocols and it can not stay tightly coupled with the specifications as required by WS-Replication.

2.3 WS-Replication

Replication is one of the main techniques to provide high availability. Paper [4] talks about the WS-Replication framework. Availability of web service is achieved with the help of replication. The same service is deployed in a set of sites called replicas, so if one site fails, the other site can continue providing the service. WS-Replication uses web service technology and SOAP for transporting information across the sites. We will be using the WS-Replication in different flavor. What differentiates is that our services can be deployed on different sites but on different protocols as well. The services can be deployed over HTTP or TCP/IP or SOAP or over JMS (Java Messaging Service) or any other protocol. The gateway will directly interact with the various services deployed on various sites to serve a client. Clients invoke a replicated web service in the same way they invoke a non-replicated web service. Internally, web service replication will take care of deploying the web service in a set of sites and the gateway will multicasting the message to replicated services and waiting for one or all of the replies and delivering single reply to the client. Various research works is going on the WS-Replication framework, so we will not discuss this in this paper.

We will consider that our services are available over different sites and we have a registry in the service gateway which categorizes all the services which take similar kind of inputs and provide the same output for a set of given inputs. These services form a community for the service gateway where client requested can be broadcasted. The services will respond back to the gateway with their responses and the gateway's components will identify and validate the responses. The first non-faulting response is sent back to the requester and the rest are ignored.

III. ENTERPRISE SERVICE GATEWAY ARCHITECTURE

The paper[2] talks about an enterprise level gateway where all the requests are sent to the gateway . The gateway verifies by examining the state of the system if the service provider is available and ready to serve the requests. This is because of the system may be available or sometimes it may not be provide the service to the users request due to security of performance reasons. It's a role of gateway that constantly monitors the different services which are provided by the enterprise. The gateway monitors the health of the services and its responsibility to forward a client request to a corresponding service. In case a particular service is not available for any reasons the gateway will decides to send the request to another service. The gateway can keep check the availability of services however it becomes a performance overhead for the gateway and can impact the processing of client requests and throughput.

The architecture presented in the paper [1] which overcomes this also it enhances the availability of web services. The service gateway is used to multicast the request over different protocols to the services which are replicated on different protocols. The enterprise service gateway will handle the following responsibilities: The client request are sent to the service gateway which holds a repository/registry of services each categorized by the kind of service provided. The gateway can multicast one request to services replicated on multiple protocols (in the service cloud).

The service gateway should also have the capabilities to transform message from one form to another that is, the gateways which can be implemented typically using enterprise service buses will have the intelligence to send the requests to the replicated services in their native protocols.

The gateway is responsible to collect the response and post it back to the service requester. This way the WSDL model need not be changed to add multiple transport capabilities as described in [1]. Enterprise service buses (ESB) like Mule, Service Mix make an excellent fit for service gateways. Most of these enterprise service buses have data transformation, message processing and data transportation capabilities over multiple protocols. We would leverage these capabilities of ESBs for the service gateway.

The service gateway comprise of message collectors, processors, collators and aggregators which help in arranging the messages in order. These tools help in collecting the request from the client, dispatching them to various services and then collating the response from various services. The gateway then

decides which response should be sent to the requestor based on the output and quality of response.

IV. EVALUATION AND PERFORMANCE

We have created service replicas on 3 different protocols such as HTTP, JMS and SOAP. Comparing the results over (Http, SOAP), (SOAP, JMS) and (Http, JMS). To implement this we have used the following tools: Apache Tomcat, Apache Active MQ and Mule. Figure 1 show how the gateway (Enterprise Service Bus) gets client request and multicasting the request to service replicas on different protocols and collects the response. Finally the responses returned back to the client. The implementation has been done in section 4.1, 4.2 and 4.3:

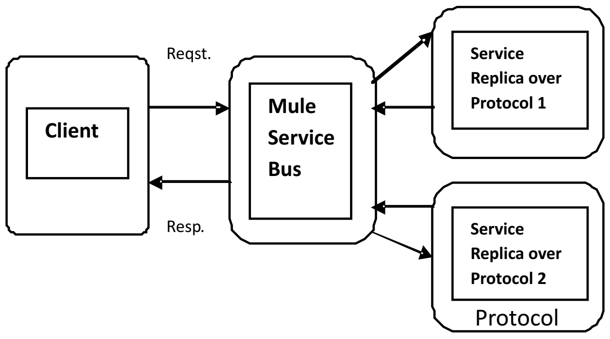
Reqst.
Client
Mule Service Bus
Resp.
Service Replica over Protocol 1
Service Replica over Protocol 2
Protocol

Fig. 1. Service Gateway (Mule) and Replication of Services on two different protocols

4.1 Service exposed over HTTP and JMS:

We had exposed one service over HTTP and another over JMS. The request from client was sent to Mule ESB. Mule forwarded the request to the services over HTTP and JMS. The services responded and the first available response was sent back to the client. The flow of request, responses is as depicted in the above figure 1. We registered the HTTP service as one of the components within Mule. Mule's request to this service have been handled synchronously and the response sent back one of Mule's endpoint which it then picked up to forward it to client.

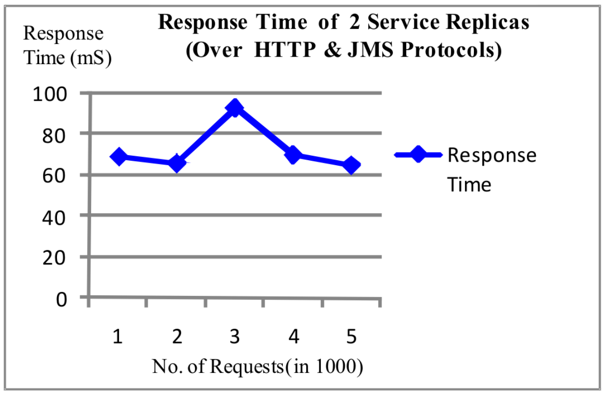

Fig. 2. Response Times when Services Replicated over HTTP and JMS Protocols

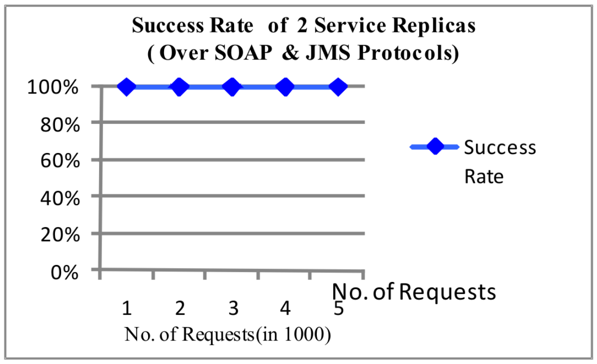

Fig. 3. Success Rate when Services Replicated over HTTP and JMS Protocols

The JMS service was available over apache active MQ infrastructure. A standalone component listening to the message queue on active MQ triggered the service picking up the request from the queue. The service responded with the response to a reply queue where Mule had a listener and it collected the response to send it back to the client. The communication over JMS was asynchronous in nature. The system responded to all requests without any failure even when one of the services was down. The above figure 2 shows the response time of requests of various sizes being serviced by the service gateway and figure 3 shows the success rate of requests of various sizes being serviced by the service gateway.

4.2 Service exposed over SOAP and JMS

For this experiment we had exposed one service over SOAP and another over JMS. The request from client was sent to Mule ESB. Mule forwarded the request to the services over soap and JMS. The services responded and the first available response was sent back to the client. The flow of request responses is as depicted in the diagram above.

We registered the SOAP service endpoint with Mule. The service itself was on a separate server deployed with Apache Axis. Mule's requests to this service have been handled synchronously and the response sent back one of Mule's endpoint which it then picked up to forward to client.

The JMS service was available over apache active MQ infrastructure. A standalone component listening to the message queue on active MQ triggered the service picking up the request from the queue. The service responded with the response to a reply queue where Mule had a listener and it collected the response to send it back to the client. The communication over JMS was asynchronous in nature. The system responded to all requests without any failure even when one of the services was down. However this infrastructure was slower in responding when compared with other setups discusses in this paper.

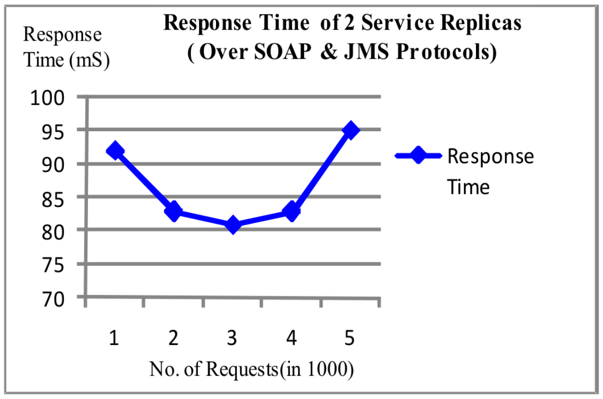

Fig. 4. Response Times when Services Replicated over SOAP and JMS Protocols

The above figure 4 shows the response time when services replicated over SOAP and JMS protocols and requests of various sizes being serviced by the service gateway and figure 5 shows the success rate of requests of various sizes being serviced by the service gateway.

4.3 Service Exposed over Soap and HTTP

For this experiment we had exposed one service over SOAP and another over HTTP. The request from client was sent to Mule ESB. Mule forwarded the

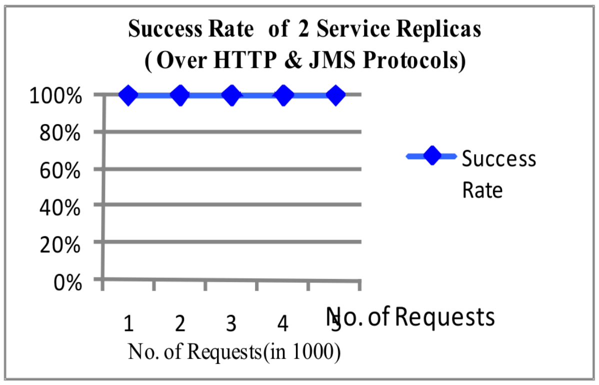

Fig. 5. Success Rate when Services Replicated over SOAP and JMS Protocols

request to the services over SOAP and HTTP. The services responded and the first available response was sent back to the client. The flow of request responses is as depicted in the diagram above.

We registered the SOAP service endpoint with Mule. The service itself was on a separate server deployed with Apache Axis. Mule's requests to this service have been handled synchronously and the response sent back one of Mule's endpoint which it then picked up to forward to client.

The HTTP service was configured as one of the components within Mule. Mule's request to this service have been handled synchronously and the response sent back one of Mule's endpoint which it then picked up to forward it to client. In this case as well the system responded to all requests without any failure even when one of the services was down. This infrastructure was the fastest in responding to all the requests made by clients.

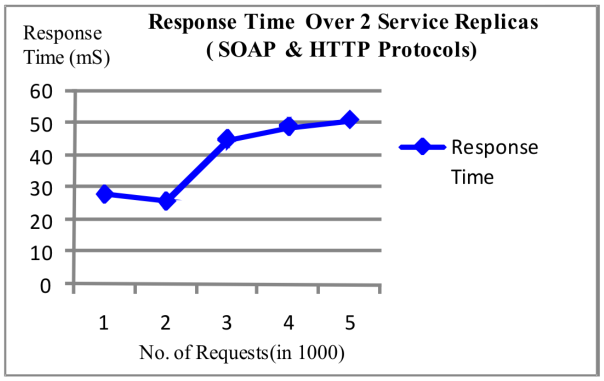

Fig. 6. Response Times when Services Replicated over SOAP and HTTP Protocols

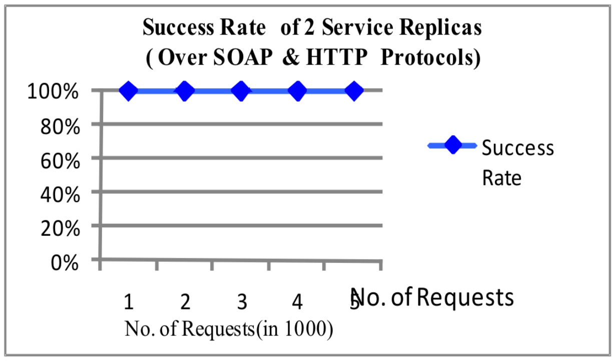

Fig. 7. Success Rate when Services Replicated over SOAP and HTTP Protocols

Thus we see that replicated services over different protocols improve the availability of the services whenever we see a risk of unavailability of a service on a particular protocol. Though the availability showed up 100% we would say that the service can be made available for 99.99% of time when replicated over different protocols.

The above figure 6 shows the response time when services replicated over SOAP and HTTP protocols and requests of various sizes being serviced by the service gateway and figure 7 shows the success rate of requests of various sizes being serviced by the service gateway.

V. CONCLUSION AND FUTURE WORK

In this paper we have implemented an architecture for mission critical applications. The central idea used here is to introduce replicated services on different protocols and have enterprise gateway control the flow of requests and response to and from the replicated services. Future work will be in the direction of checking the load that this type of service replication can handle. Load testing the architecture will set the guidelines about the scalability of the architecture as a whole.

REFERENCES

[1] Subil Abraham, Mathews Thomas , Johnson Thomas," Enhancing Web Services Availability" *Proc 26th IEEE International Conference on Software Engineering*, 2005

[2] Ms. R. Sundharam, M. Lakshmi,D. Abarajithan, "Enhancing the High Availability of Web Services for Mission Critical Applications". International Conference on Trendz in Information Sciences and Computing, (TISC 2010),IEEE – TISC 2010.

[3] Ms. R. Sundharam, M. Lakshmi, D. Abarajithan, “Enhancing the High Availability of Web Services with Replicated Services on Different Protocols”. International Conference on Distributed Computing Engineering (ICDCE 2011).

[4] Jorge Salas, Francisco Perez-Sorrosal, Marta Patino-Martinez and Ricardo Jimenez-Peris “ WS-Replication: A Framework for Highly Available Web Services”, International World Wide Web Conference Committee (IW3C2).WWW 2006,Edinburgh, Scotland, ACM 1-59593-323-9/06/0005.

[5] Sattanathan Subramanian. “High-Availabe Web Service Community System”. In the Proceedings of The Sixth Iinternational Conference on Information Technology: New Generation, 2009.

[6] Evan March and Hal Stern, *Blueprints for High Availability*, Wiley Publishing Inc, 2003.

[7] K. Birman, R. Van Renesse, W. Vogels, “Adding High Availability and Autonomic Behavior to Web services”, *Proc International Conference on 26th IEEE on Software Engineering*, 2004.

[8] WS-Reliable messaging - http://www-128.ibm.com/developerworks/webservices/library/specification/ws-rm/

[9] H. Maruyama. New trends in e-Business: from B2B to web services. New Generation Computing, 20(1), 2002.

[10] Q.B. Vo, M. Huhns R. Kowalczyk and Z. Maamar. Introduction. Special issue on Service-Oriented Computing: Agents, Semantics, and Engineering in the International Journal on Agent-Oriented Software Engineering, 2009 (Inpress).

[11] Floyd Piedad, Michael Hawkins, *High Availability: Design, Techniques and Processes,* Prentice Hall, 2000.