Volume 2, Issue 5, May 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Implementation of Middleware Services Based on the Servers for Applications

P Karthikeyan*, Dr. E Sathiyamoorthy
School of Information Technology & Engineering,
VIT University,India
mailme_kk@yahoo.com

*Abstract— **The Service Level Agreement (SLA) is a document that defines and identifies the various services given by a service provider to its clients. SLA specifies the various metrics such as service availability, service efficiency, service throughput, etc. The manual Service Level Agreement is eliminated with the use of the middleware architecture which dynamically allocates the resources to the services. The middleware generates an SLA document based on the usage of the services which results in the popularity of the services. Our architecture for SLA is implemented using the open source application server JBoss and using the Eclipse platform. JBoss is allowed to meet the services of the applications it hosts i.e., to honor the SLAs of the applications. In this paper it is shown how to take advantage of the middleware, the prototype of the middleware services using JBoss Application Server and implement such that the applications that support will all be able to honour SLA.***

Keywords— *Service Level Agreement (SLA), middleware architecture, Qos requirements, Application Server, Cluster*

I. INTRODUCTION

An agreement between the customer and the service provider is known as the Service Level Agreement (SLA). This agreement was done manually before. The level of service for this contract is altogether formally defined. The main requirements of this SLA are service availability, service efficiency and service throughput. There are many segments that the SLAs include to address. Definition of services, performance measurement, problem management, customer duties, warranties, disaster recovery and termination of agreement are some of them. All these segments are used to maintain and help the improvement of the services provided by the SLAs. To eliminate the manual process of the Service Level Agreement middleware architecture is used. The middleware architecture allocates the resources dynamically to the various services available which is called clustering. The clustered resources are allocated only to the hosted applications i.e. at the run time. There are number of nodes available for the resources that are clustered. And they are assigned to the various services accordingly for the hosted application. The middleware also load balances the requests from the client ensuring that no particular node is overloaded with many requests at a time. The middleware also detects the node failures from the cluster thus providing high availability of the Application Server in a cluster. The middleware generates an SLA document based on the popularity of the services provided. Our architecture for SLA is implemented using the open source application server JBoss and using the Eclipse platform. JBoss is allowed to meet the services of the applications it hosts i.e., to honor the SLAs of the applications. In this project taking advantage of the middleware and the prototype of the middleware services using JBoss Application Server is implemented such that the applications that support will all be able to honor SLA.

II. RELATED WORK

In *Aggregation of Service Level Agreements* in the Context of Business Processes, there is a method defined as to how a service provider can aggregate the SLAs of the individual services within a business process into a single SLA [7]. The future research was yet divided into three parts for future work. They were to improve the existing aggregation algorithms by including BPEL features left out in the current version, to examine the runtime behavior of different process engines in order to consider it during the aggregation, and to extend the current approach for coping with more complex message exchange patterns. In *the Role of Service Level Agreements in Classified Advertisement Websites,* the Service Level Agreements (SLAs) plays an important role for maintaining the Quality of Service [6]. It helps for the improvement of the level of services that are being offered. The level of services is offered for the service consumers for the validation of the services that are received. But still it was just a work-in-progress and the authors were implementing the architecture described there.

III. EXISTING SYSTEM

The existing system of SLA enforcement for SOA (Service Oriented Architecture) Applications using a middleware in figure 1.1(refer "Adaptive SLA Enforcement for SOA Applications using Middleware"[1]) dynamically allocates the clustered resources to the services and generates the SLA document based on the popularity of the services deployed in the cluster. It allocates the resources to the hosted applications. It is done dynamically i.e. at the run time unlike it was done before, manually. The existing system mainly provides the middleware services to the Application Server. And thereby enables the server for the understanding of the service requirements of the applications that are hosted.

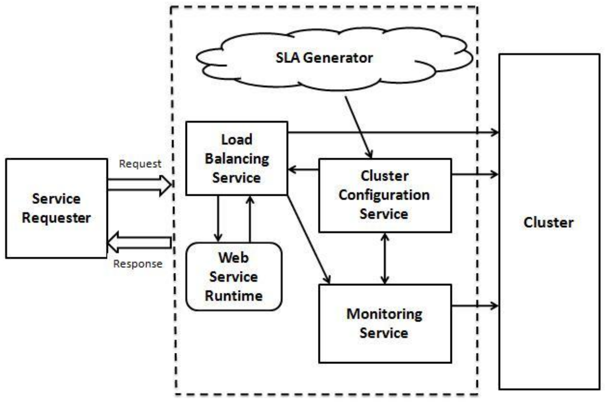

Figure 1.1

IV. PROPOSED SYSTEM

The proposed system of SLA enforcement for SOA Applications using the middleware architecture involves dynamic resource management even when multiple applications are concurrently deployed in the server cluster. It uses "ALTERNATE RESOURCES ALLOCATOR" dynamically to allocate resources for more number of Applications. The advantage of this system is that the middleware is tested and found to be acquiring the SLA driven resource provision based on the servers. Cluster configuration is also done effectively to provide for the demanded services made by the client. Hence the client is always satisfied with since the service requirements are met by the application server cluster.

The *cluster configuration service* configures the cluster at the web service deployment time and possibly reconfigures the cluster at the runtime. Since the change in the configuration of one service will reflect the change for the remaining services of SLA parameters too, a *logger service or monitoring service* is designed to maintain the details and the history of the SLA specifications for each service deployed in the cluster. *Load Balancing Service* balances the load of client requests among the clustered nodes. It helps in avoiding overloading any particular node or sending requests to nodes which may not be available at the moment. The *alternate resource allocator* is used to provide resources for all the hosted applications based on the servers dynamically without disappointing the client.

The new module is generated and its parameters should be accurate for the reliability of the SLA. The SLA agreement generating module must be available for the SLA generation to be unhindered during the implementation of the services. In case of a failure in a module it must have an alternative one to further process the SLA. If the server is offline then an alternative server must be provided for it to further function properly.

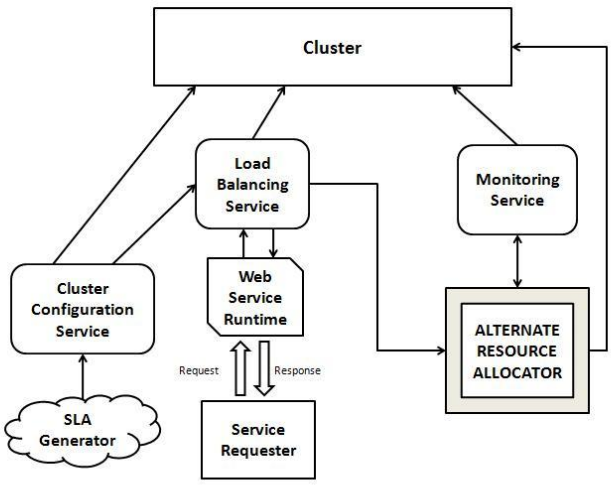

Figure 1.2

A. *What is an alternate resource allocator?*

The "Alternate Resource Allocator" is the new module which is included with this architecture to allocate the resources to the hosted applications dynamically. The advantage of this alternate resource allocator is it supports the resources to be distributed for more than one application at a time from a single server. It helps in allocating the resources to the various applications accordingly. All the modules included in this paper are described below.

B. *Modules*

1) *SLA Generation & Monitoring:* Whenever the popularity of the services changes new cluster is formed and SLA is generated. The SLA generator computes the maximum resources and the popularity factor to be allocated for the services as it is seen in figure 1.3. The resources must be optimally allocated for the services in the clustered. Individual resource allocation depends on the resources that have been allocated for other services in the cluster. So the changes made in resource allocation of a particular service will reflect changes in SLA parameters of other services. The history of SLA specifications is maintained by a monitoring service called *logger service* as seen in figure 1.4. Modifications in SLA document leads to changes in cluster configuration.

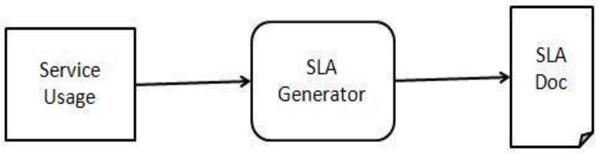

Figure 1.3

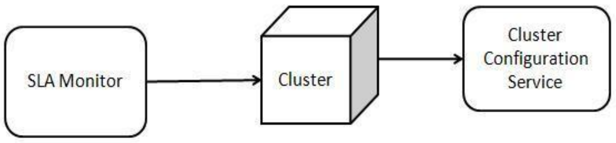

Figure 1.4

2) *Cluster Configuration Service:* Cluster is configured at deployment time of web services. Cluster reconfiguration is done during run time. Initially cluster is formed using the minimal set of available nodes; default SLA parameters are used for this purpose as seen in figure 1.5. Reconfiguration of clusters occurs under these conditions. Such as cluster node failure and replacement, spare node maintenance and increased load on clusters. Nodes within the cluster may also be released when needed. Resources which have been allocated for the released node must be de-allocated and added to the pool of spare nodes.

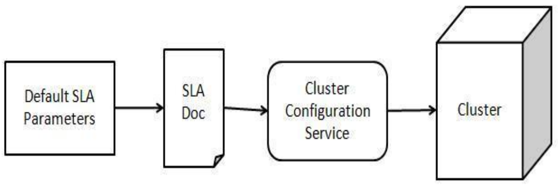

Figure 1.5

3) *Load Balancing Service:* Load balancing service balances SOAP client request. The purpose of load balancing services are avoiding overloading a node with more number of services, helping high availability of resources, emulates client for making new request and uses Web Service Definition Language (WSDL) for extracting technical information and creates SOAP requests to that web service. Figure 1.6 shows how the load balancing service also allows the cluster configuration service to update the changes.

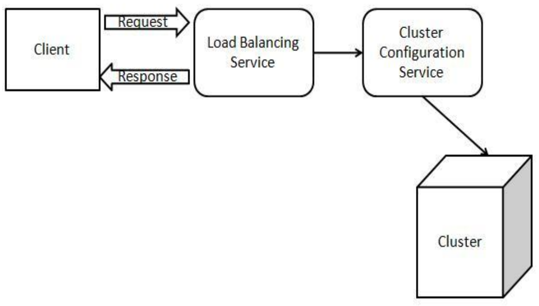

Figure 1.6

4) *Alternate Resource Allocator:* Whenever a node joins a cluster, it requests for resources. The load balancing service allocates the resources. If the resources become unavailable in critical situations an alternative resource must be provided in a negligible period. This alternative resource allocator takes cares of these abnormal situations and increases customer satisfaction without even the awareness of the client. The alternate resource allocator makes sure that the resources are allocated to the applications hosted at the run time and figure 1.2 shows that above.

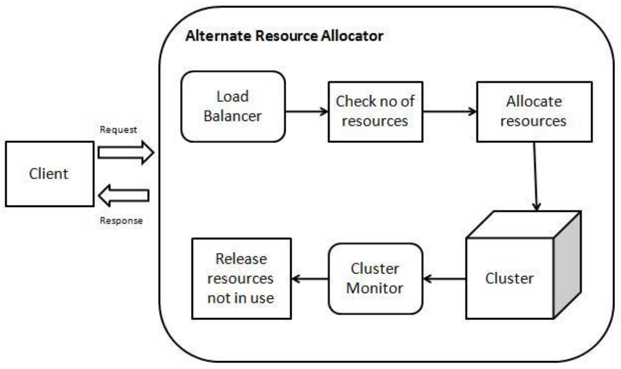

Figure 1.7

V. CONCLUSION AND FUTURE WORK

The existing load-balancing algorithms are replaced with other adaptive load-balancing methods. They are aware of the runtime load conditions of the node based on the applications and the servers. In this the client will always be satisfied since the main requirements of the client are met by the application server cluster. The application server performance is enhanced by the load balancing between the various servers of the cluster.

The extension of the work can be to deploy the QoS clustering mechanisms in emerging application areas such as the Grid computing areas. Our future work will also include adopting as many servers as it could for numerous applications to be hosted at run time and to be clustered dynamically without any constraints. Even multiple server crashes issues can be looked into for further investigation in the future.

REFERENCES

[1] IEEE. (2010) *Towards a Methodology to Define Service Level Agreements in a Convergent Network Scenario*, Luis Guillermo Martínez Ballesteros, Researcher, Engineering School, Universidad Sergio Arboleda, Bogota, Colombia lgmartinezb@ieee.org, luisgui.martinez@usa.edu.co.

[2] *Enabling Self-Organising Service Level Management with Automated Negotiation* (2010), Lei Liu; Schmeck, H.; Karlsruhe Inst. of Technol. (KIT), Inst. AIFB, Karlsruhe, Germany

[3] *Design Service Level Agreements in Outsourcing Contracts* (2010), Jinmei Huai, School of Economics and Management, East China Jiaotong University, Nanchang, China, fu30061@sina.com.

[4] *Design Service Level Agreement Based on Dynamic IT* (2009), Jinmei Huai; Sch. of Econ. & Manage., East China Jiaotong Univ., Nanchang, China

[5] *An Investigation of the Role of Service Level Agreements in Classified Advertisement Websites* (2009), Soomro, A. Song, W. Yang Li, Dept. of Comput. Sci., Univ. of Durham, Durham, UK

[6] *Aggregation of Service Level Agreements in the Context of Business Processes* (2008), Tobias Unger, Frank Leymann, Stephanie Mauchart, Thorsten Scheibler University of Stuttgart Institute of Architecture of Application Systems (IAAS) Universit¨atsstrasse 38, 70569 Stuttgart, Germany funger, leymann, mauchart, scheiblerg@iaas.uni-stuttgart.de

[7] *IJARCS, Adaptive SLA enforcement for SOA Applications Using Middleware* (2010), Prof.P.Karthikeyan, Prof.E.Sathya Moorthy and S.Senthil Ganesh School of Information Technology, VIT University, Vellore, India mailme_kk@yahoo.com