ISSN:0975-9646



# Analysing Excellence Measure, Recital of Real World Web Services

D.Venkatesh[1] & Mr.C.Sivaraman[2]
[1]*PG Scholar, Department of Computer Science and Engineering*
[2]*Assistant Professor, Department of Computer Science and Engineering*
*Bharath University, Chennai*

**Abstract-Monitoring Quality of Service (QoS) compliance is an important procedure in web service environment. It determines whether users' prospects are met, and becomes the vital factor for them to decide whether to continue paying for the service or not. The monitoring is performed by checking the actual services performance against the QoS stated in Service Level Agreement (SLA). In relation to that, the need for monitoring vague QoS specifications in SLA has become more apparent nowadays. Providing ability for monitoring QoS that is specified vaguely in SLA could give new insights and implications to web services field. This paper concludes with some recommended future works to construct the theory and perform the empirical research.**

**Keywords:QoS(Quality of Service, Service Level Aggrement.**

## 1. INTRODUCTION

Generally, data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be
used to increase revenue, cuts costs, or both. Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases. Web services have been promising in recent years and are by now one of the most popular techniques for building distributed systems. Service-oriented systems can be built efficiently by dynamically composing different web services, which are provided by other organizations. The quality-of-service (QoS)-oriented systems are highly reliant on the quality of employed web services. The main scope of our proposed work is to solve the above privacy issue on finding QoS in web services, we propose Service evaluation technique. More specifically, we utilize QoS has become an important differentiating point of different functionally equivalent web services. Web service QoS includes a number of properties, such as response time, throughput, failure probability, availability, price, popularity, and so on. Values of server-side QoS properties (e.g., price, popularity) are usually advertised by service providers and identical for different users.QoS is widely employed for describing non-functional characteristics of web services. Still a lack of comprehensive real-world web service QoS data sets for validating various QoS-based approaches. With the increasing number of web services, QoS has become an important differentiating point of different functionally equivalent web services. However, without comprehensive real-world evaluations, sufficient web service QoS values cannot be collected. It is thus difficult to validate the feasibility and effectiveness of various QoS-based approaches in service computing.
Traditional web service search approaches typically only exploit keyword-based search techniques without considering QoS of web services. In reality, web services sharing similar functionalities may have very different no functionalities. To effectively provide personalized web service search results to different users, it is requisite to consider both functional and nonfunctional characteristics of web services when searching web services.

## 2. RELATED WORK

The cloud server in our work is considered as "honest-but-curious" , a model extensively used in SSE and characterized by that the cloud server will honestly follow the designed protocol but is curious to analyze the hosted data and the received queries to learn extra information.

### 2.1 Statistic leakage

Although all data files, indices and requests are in encrypted form before being outsourced onto cloud, the cloud server can still obtain additional information through statistical analysis. We denote the possible information leakage with statistic leakage. There are two possible statistic leakages, including term distribution and inter distribution. The term distribution of term t is t's frequency distribution of scores on each file $i(i \in C)$. The inter distribution of file f is file f's frequency distribution of scores of each term $j(j \in f)$. Term distribution and inter distribution are specific [10]. They can be deduced either directly from ciphertext or indirectly via statistical analysis over access and search pattern [8]. Here access pattern refers to which keywords and the corresponding files have been retrieved during each search request, and search pattern refers to whether the keywords retrieved between two requests are the same. Based on our observation, distribution information implies similarity relationship among terms or files. On one hand, terms with similar term distribution always have simultaneous occurrence. For instance, obviously, the term "states" are very likely to co-occur with "united" in an official paperwork from the White House, and their term distribution, not surprisingly, are very same in a series of such a kind of paperwork. Given these paperwork are encrypted but term distribution are not concealed, so once an adversary somehow cracks out the plaintext of "united", he can reasonably guess the term that shares a similar term distribution with “united” may be “states”. On the other hand, files with similar inter distribution are always the same category, e.g., two medical records from a dental surely are the same category, and they are very likely to share a similar inter distribution. Therefore, this specificity should be hidden from an un trusted cloud server.

Table 1: Similarity relevance with “resources” before and after OPM

| term | $len_t$ | $\kappa$ | $len'_t$ | $\kappa'$ |
|---|---|---|---|---|
| directorate | 264 | 0.023 | 264 | 0.023 |
| education | 4826 | 0.544 | 3573 | 0.403 |
| human | 7648 | 0.885 | 7647 | 0.885 |
| provide | 1014 | 0.098 | 1014 | 0.098 |
| sciences | 2480 | 0.226 | 2480 | 0.226 |

**2.2 κ-similarity relevance**

In order to avoid information leakage in server-side ranking schemes, a series of techniques have been employed to flatten or transfer the distribution of relevance scores. These approaches, however, only cover the distribution of individual term or file, ignoring the relevance between them and the violation of data privacy that arouses thereafter. In order to formulate this problem, we propose the concept of κ-similarity relevance.

## 3. SYSTEM MODEL

In this section, our system model is designed with two phases. In Phase 1 consists of analysis and design is considered and in Phase 2 the model is implemented and validated. The Initial analysis activity involves researching the problem in detail and evaluating various approaches to resolve the same.

Design activity is carried out in two fold, high level and low level design. High level design involved architectural design of the framework and case study planning as it plays a vital role in validating the approach.

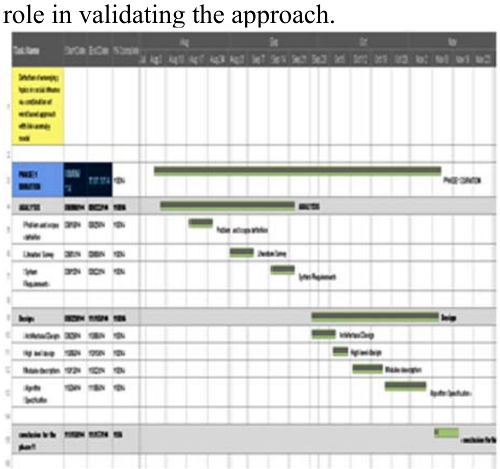


Fig.2.1 Design Activity

Web service composition enables seamless and dynamic integration of business applications on the web. The performance of the composed application is determined by the performance of the involved web services. Therefore, non-functional, quality of service (QoS) aspects (e.g. response time, availability, etc.) is crucial for selecting the web services to take part in the composition. The problem of identifying the best candidate web services from a set of functionally-equivalent services is a multi-criteria decision making problem. The selected services should optimize the overall QoS of the composed application, while satisfying all the constraints specified by the client on individual QoS parameters.

## 4. ARCHITECTURAL OVERVIEW

System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behaviour) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system.

Fig.4.1 SYSTEM ARCHITECTURE

## 5. AUTHENTICATION

In this section, we first discuss the authentication of with user name and password. With that, we give some preliminaries about monitoring by checking the actual services performance against the QoS stated in Service level agreement (SLA) and privacy level agreement (PLA), followed with the preliminaries of trust and reputation and the preliminaries of trusted center entity (TCE).

In authentication module is used to check the user as valid or invalid. In this module, when we enter the username and password, this username and password is check into the database. If username and password is correct then allow to next processing, otherwise it consider as invalid user and again go to the login process otherwise generates an error message with the login page.

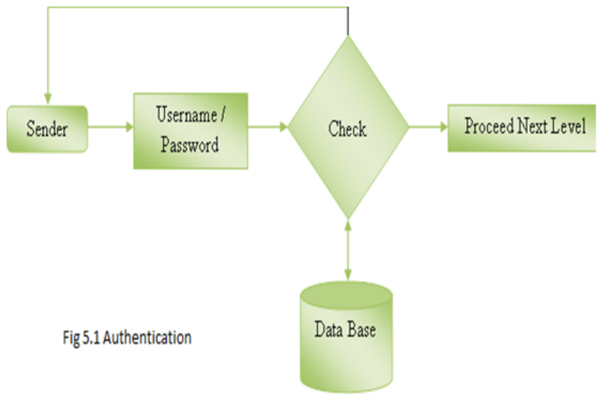



Fig 5.1 Authentication

### 5.1 SERVICE PROVIDER

It does not specify how the service is engineered or provisioned; it simply specifies what is delivered to the end user. How the service provider decides to construct the service infrastructure is not part of the Service Definition. Service provider can collect the consumer need according to their sopaecification.web service provider will find the QOS functions for all web services.

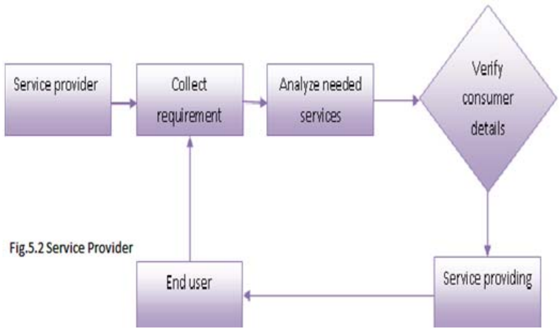



Fig.5.2 Service Provider

.

## 6. DATASET COLLECTION

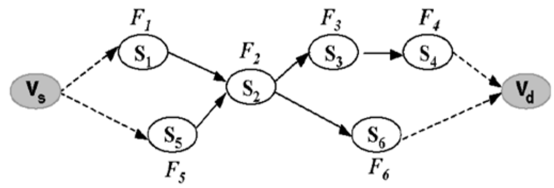
In this module we describe dataset collection for multiple real world web services. The term "dataset" originated in the mainframe field. A data set (or dataset) is a collection of data, usually presented in tabular form. Each column represents a variable. Each row corresponds to a member of the data set. It lists values for each of the variables. The data set may comprise data for one or more members, corresponding to the number of rows.

### 6.1 Given Input & Output Design:

Authentication
Input: Username and password
Output: valid or invalid
SERVICE PROVIDER
Input: collect web service dataset
Output: analyze web service properties

### 6.2 Technique Used

Web service evaluation is a main approach for obtaining QoS values. Conducting web service evaluation is difficult, since real-world web service invocations consume resources of service providers and impose costs of service users. The evaluation focuses on investigating the response time and throughput performance of web services. Response time is defined as the time duration between a service user sending a request and receiving the corresponding response, while throughput is defined as the average rate of successful message size (here in bits) delivery over a communication channel per second




We begin by using a simple example to show the process of service composition with end-to-end
QoS constraints. Figure 1(a) shows a composite service that has four possible paths: $\{F1, F2, F3, F4\}$, $\{F1, F2, F6\}$, $\{F5, F2, F3, F4\}$, or $\{F5, F2, F6\}$. Each function $Fi$ may be executed by any
atomic service ($si\ j$ ) in the service class $Si$ . The QoS values of each atomic service are shown in
Figure 1(b). The end-to-end QoS requirements include:
Response time $\leq 600$,
Cost $\leq 250$,
Availability $\geq 85\%$.
The optimal selection of services, which maximized the sum of all service utilities, is $\{s11, s21, s31, s42\}$ with a total utility of 823, a response time of 590, at a cost of 240 and an availability of
86.64%.
In this simple example, there are a total of 30 candidate combinations, each with different utility
and QoS values. The number of possibilities goes up exponentially with an increasing number of
service classes. The complexity of service composition is due to the following factors.

## 7. CONSLUSION

Conducts evaluations on user-observed QoS of web services from distributed locations..A large number of web service invocations are executed by service users under heterogeneous environments on real-world web services. Comprehensive experimental results are presented and reusable data sets are released. Besides failure probability, response time, and throughput, more QoS properties being investigated.

## REFERENCES

[1]. Zibin,Zheng,Member,IEEE,Yilei Zhang, Student, Member,IEEE,and Michael R.Lyu,Fellow,IEEE "Investigating QOS of Real World Web Services-IEEE TRANSACTIONS ON Services Computing ,Vol,No .1,January -March 2014.

[2] C.-L. Fang, D. Liang, F. Lin, and C.-C. Lin, "Fault-Tolerant Web Services," J. System Architecture". VOL. 5, APRIL JUNE 2012

[3] Zibin Zheng, Student Member, IEEE, Hao Ma, Michael R. Lyu, Fellow, IEEE, and Irwin King, Senior Member, IEEE "QoS-Aware Web Service Recommendation by Collaborative Filtering" VOL. 4, NO. 2, APRIL-JUNE 2011

[4] Z. Zheng and M.R. Lyu, "A Distributed Replication Strategy Evaluation and Selection Framework for Fault-Tolerant Web Services" Sept. 2008, L.-J. Zhang, J. Zhang, and H. Cai, Services Computing. Springer, 2007.

[5] X. Chen, Z. Zheng, X. Liu, Z. Huang, and H. Sun, "Personalized QoS-Aware Web Service Recommendation and Visualization," IEEE Trans. Services Computing, vol. 6, no. 1, pp. 35-47, 2011.
[6] Z. Zheng, H. Ma, M.R. Lyu, and I. King, "QoS-Aware Web Service Recommendation by Collaborative Filtering," IEEE Trans. Service Computing, vol. 4, no. 2, pp. 140-152, Apr.-June 2011.
[7] Z. Zheng, H. Ma, M.R. Lyu, and I. King, "Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization," IEEE Trans. Service Computing, vol. 6, no. 3, pp. 289-299, July-Sept. 2013.
[8] M. Alrifai and T. Risse, "Combining Global Optimization with Local Selection for Efficient QoS-Aware Service Composition," Proc. 18th Int'l Conf. World Wide Web (WWW '09), pp. 881-890, 2009.
[9] L. Zeng, B. Benatallah, A.H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-Aware Middleware for Web Services Composition," IEEE Trans. Software Eng., vol. 30, no. 5,pp. 311-327, May 2004.
[10] C.-L. Fang, D. Liang, F. Lin, and C.-C. Lin, "Fault-Tolerant Web Services," J. System Architecture, vol. 53, no. 1, pp. 21-38, Jan. 2007.
[11] Z. Zheng and M.R. Lyu, "A Distributed Replication StrategyEvaluation and Selection Framework for Fault-Tolerant Web Services," Proc. Sixth Int'l Conf. Web Services (ICWS '08), pp. 145-152, 2008.
[12] Z. Zheng and M.R. Lyu, "A QoS-Aware Fault Tolerant Middleware for Dependable Service Composition," Proc. 39th Int'l Conf. Dependable Systems and Networks (DSN '09), pp. 239-248, 2009.