A QoS Evaluation Method for Personalized Service Requests

Rutao Yang[1, 2], Qi Chen[1, 2], Lianyong Qi[1, 2], Wanchun Dou*[1, 2]

[1] State Key Laboratory for Novel Software Technology, Nanjing University

[2] Department of Computer Science and Technology, Nanjing University

210093, Nanjing, China

yrutao@gmail.com, adios737@126.com, lianyongqi@gmail.com, douwc@nju.edu.cn

Abstract. With the prevalence of Web service, QoS is playing a more and more important role in service evaluation, recommendation and selection. In most previous works, it is often assumed that the delivered QoS of a Web service is often determined by service provider, not service consumer. However, in the practical service execution environment, Web services usually work in an interactive mode with service consumer, so service consumer should also take responsibility for the delivered QoS of a Web service. Hence, it becomes a challenge to evaluate the QoS of Web services impartially. In view of this challenge, a QoS evaluation method for personalized service requests is proposed in this paper. Finally, the effectiveness of our method is validated, and an optimization method is proposed to improve the QoS evaluation efficiency.

Keywords: QoS evaluation, Service request, Collaborative Filtering, Clustering

1 Introduction

Increments of personalized software requirements and the adoption of partial software outsourcing ideology bring new challenge for the traditional commercial off-the-shelf software solution. The Software as a Service (SaaS) model has become a possible solution by the widespread availability of fast Internet access, combined with widespread acceptance of service oriented architecture (SOA). However, when the number of software services，that are delivered on-demand and priced on-use, has expanded dramatically, it becomes a challenge to select appropriate services for service requesters among many functionally equivalent services. Hence, evaluating the quality of service (QoS) [1][2] plays an important role in service recommendation, selection, and composition. Usually, the QoS published by service providers are distrustful, because QoS is variable during different service invocations and providers may give inauthentic high quality information in order to attract more potential service consumers. In this circumstance, the reputation of service providers or ratings of services are mined based on historical execution QoS information. However, bad execution QoS is usually all ascribed to service provider's responsibility unfairly. A Web service is seen as an application accessible to other applications over the Web [3], which means services are provided in an attractive mode. Then personalized service requests will cause the variation of the service's delivered QoS. Most researchers pay more attention to the QoS evaluation from provider's perspective, and ignore the effect from consumer's perspective.

In view of the issue referred above, we put forward a QoS evaluation method for personalized service requests. Specifically, a service request model is proposed to specify personalized requirements, and then candidate services' quality are evaluated based on collaborative filtered historical execution

information. The remainder of the paper is organized as follows: Section 2 discusses the motivation of this paper. Section 3 demonstrates our proposed personalized QoS evaluation method, followed by performance analysis and optimization of the method in Section 4. In Section 5, the related work is discussed. Section 6 concludes the paper.

2 Motivation

As explained before, QoS evaluation is an important step before service ranking, recommendation, selection and composition. Once the service provider-consumer relationship is potentially established, it usually refers to three kinds of QoS, which are published QoS (promised QoS), required QoS (expected QoS) and execution QoS (delivered QoS). They are time consecutive, and refer to three procedures of SOA (see Fig. 1). Along with a service is registered, QoS information is published to demonstrate the service's quality. Then consumers request a service with QoS constraints, and lastly consumers usually give feedback for the execution QoS after invoking a service. By the way, execution QoS information is monitored to record in historical log for helping service providers to improve service's quality. Published QoS is directly used for service selection or composition in some literature (say [4]). The trustworthiness of published QoS is ignored or published QoS is assumedly credible. However, in practice, the execution QoS cannot be known until the service is invoked and QoS evaluation is a necessary prelude.

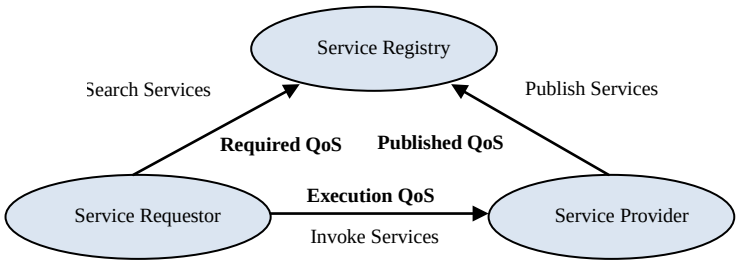

Fig.1. Service Oriented Architecture (SOA)

The trustworthiness of published QoS is commonly measured by the service's reputation. As proposed in previous work, the reputation either depends on the ratings of user's experience [5], or is computed based on the actual measurement of the conformance of execution QoS to promised QoS [6]. However, the former solution suffers fairness problem of user ratings of services, especially in case malicious consumers may give false ratings and subvert services' reputation. And the later solution suffers from responsibility impartiality problem because of services are performed in an interactive mode. Consumers' network environment, personalized service request and payment will affect execution QoS. And empirical results show that different consumers likely have different experiences (e.g., Failure Probabilities) of invoking the same service (see Fig. 2). However, similar consumers' historical execution QoS can make a good contribution to the prediction of future execution QoS.

Based on the motivation discussed above, a personalized QoS evaluation method is proposed in this paper. Firstly, a service request model is defined for specifying personalized service requirement. Secondly, a collaborative filtering process of historical execution QoS is put forward. Then, the left records in previous step are used to compute the estimated execution QoS. Finally, the effectiveness of our method is validated, and an optimization method is proposed to improve the QoS evaluation efficiency.

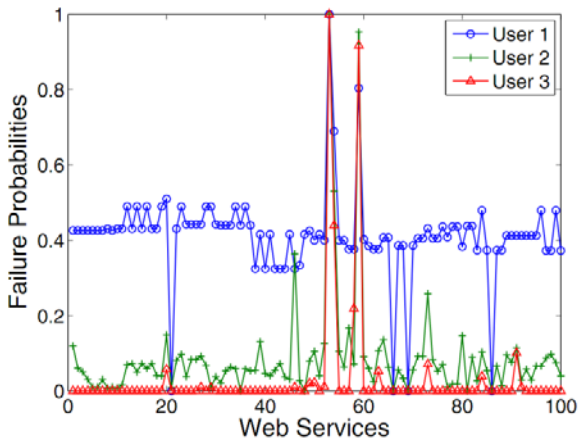

Fig.2. Different users have different failure probabilities (from [7])

3 A QoS Evaluation Method for Personalized Service Requests

The goal of personalized QoS evaluation is to provide a fair and appropriate QoS evaluation in user-centric manner for service selection and recommendation.

3.1 Service Request Model

Usually, a service request contains functionality requirement and QoS constraints. In most work of QoS-aware service selection and composition, functionality requirement are used to discover candidate services that are functionally equivalent. However, the scale of input data or invoking times of a service will affect the execution QoS, especially execution duration. And the input data or invoking times are variable depending on consumers' requests. So, data input is extracted from traditional functionality requirement for specifying personalized service request, and the functionality still describes the ability of performing some computing task.

Fig.2 shows that different users have different failure probabilities even when they invoke the same service. This phenomenon is caused by different context scenario, which characterizes the situation of a user, place or the interactions between users, applications and the environment [8]. As once a service request is delivered to a service provider, the main work responding the request is offered by the service. The interactions between service consumers and service providers are network environment, which affect the time in request delivering and result returning, especially while the service is about multimedia or large scale data processing. So, service requests are encouraged to contain client-side network environment, which can be measured by bandwidth, or other criterion.

As software services are priced on-use, payment should obviously be contained in a service request, and also payment is a main part of service level agreement (SLA). The reputation is a special criterion for evaluating a service, as it reflects the trustworthiness of a service provider (or published QoS), and then indirectly reflects a service's quality. So, in our service request model, the reputation is separated from other service quality, which is also used in literature [6].

Based on above discussion, a service request model is defined as follows:

Definition 1(Service Request Model): A service request is a tuple {*Functionality Description*, *Input Data*, *Network Environment*, *Price*, *Reputation*, *Quality Constraints*}, where *Quality Constraints* is also a tuple {*response time*, *availability*, *failure probability*}.

Specifically, *Network Environment* is not provided by service requester explicitly, but it can be evaluated based on the happening time and location of service request to mine network congestion. It's worth noting that *Quality Constraints* can be expanded by other QoS criteria. Take an online data mining software service for example, an instance of service request can be specified as {Mining task, 1G data, 100M bandwidth, 2$, above 9.0 (reputation's unbound is 10), {less than 10s, 95%,95%}}.

3.2 User-centric Collaborative Filtering

QoS evaluation is indeed a prediction of delivered QoS in future service invoking. Machine learning and reasoning under uncertainty have generated a variety of techniques that fall under the umbrella of predictive statistical models in Artificial Intelligence areas. Two main approaches have been adopted to perform prediction task: content-based and collaborative [9]. In the application of QoS evaluation, the former approach requires that the service requester has ever invoked the service in history, which is usually unrealistic. So, collaborative approach, in which the QoS of a service is predicted from the behavior of other like-minded service invoking, is employed in this paper. As one of the most successful approaches to building recommender systems, collaborative filtering [10], in our QoS evaluation, uses the known QoS of a group of history users to make predictions of the unknown execution QoS for a new user.

As explained in Section 2, a collaborative filtering process is necessary for evaluating a service impartially in user-centric manner. Here the collaborative filtering is based on the factors that affect different users' experience QoS. As specified in **Definition 1**, functionality description is used for discovering candidate functionally equivalent services. Reputation and QoS constraints are used to service matchmaking. And the left elements in service request model, which can be demoted as a tuple named as *specialRequest*{*input*, *network*, *price*}, are the basis for collaborative filtering. The processing target of collaborative filtering is the historical records of service invoking. And the task of collaborative filtering is to identify the similar service requesters based on their similarity with a given *specialRequest*. Each service requestor' invoking information, demoted as *aRequest*{*input*, *network*, *price*}, can be extracted from its historical record. The similarity between *specialRequest* (*sR*) and *aRequest* (*aR*) is computed as follows:

$$similarity(sR, aR) = 1 - \frac{length(sR, aR)}{\sqrt{n}} \tag{1}$$

$$length(sR, aR) = \sqrt{\sum_{i=1}^{n} \left(\frac{|sR[i] - aR[i]|}{Max[i] - Min[i]} \right)^2} \tag{2}$$

Here n=3, as both *sR* and *aR* has three dimensions. *sR*[*i*] and *aR*[*i*] denotes the value of i*th* dimension of *sR* and *aR*. *Max*[*i*] and *Min*[*i*] denotes the maximum value and minimum value of i*th* dimension. The above formulas show that the *similarity* of *sR* and *aR* is based on their *length*, and the *length* is the normalized Euclidean distance. It's a general knowledge that the lower of their distance, the bigger of their *similarity*. The more *aR* is similar to *sR*, the more accurate (weight) its executive experience is supposed to be as a QoS prediction in next step. When a historical request' similarity with the new request is beyond of a threshold value, the corresponding invoking QoS information is filtered out, as it has less reference to the specialized QoS evaluation. The threshold value as a system parameter determines the number of left historical records. However, even the threshold value is set the maximum value, i.e., *sqrt*(*n*), the personalized QoS evaluation can also work on, because of the existence of weight, i.e., similarity, for each record.

3.3 Computing QoS Prediction

QoS evaluation can be categorized as the predictive statistical problem. And Section 3.2 has produced the filtered historical records set *S* and the weights *W* of these records for QoS prediction. It's worth noting that a provider's QoS constraints are usually offered as an interval, and the candidate services for personalized QoS evaluation have satisfied the service matchmaking. As explained in Section 3.1, the criteria for personalized QoS evaluation contains *price*, *quality* {*response time*, *availability*, *failure probability*}, and *reputation*. All of these QoS criteria can be combined in a tuple *q* with size of 5, i.e., {*price*, *response time*, *availability*, *failure probability*, *reputation*}.

In mathematical statistics area, mean value and variance are two important criteria to represent estimation result. The corresponding formulas for computing QoS estimation in our method are denoted as follows:

$$E(q_i) = \frac{\sum_{i=1}^{m} w_i R_i}{\sum_{i=1}^{m} w_i} \tag{3}$$

$$V(q_i) = \sum_{i=1}^{m} w_i^2 \left(R_i - E(q_i)\right)^2 \tag{4}$$

where $E(q_i)$ and $V(q_i)$ are the mean value and variance of QoS criterion q_i, and m is the number of records, w_i is the weight of record R_i. It should be noted that the value of failure probability in each invoking record is 0 or 1.

As discussed in this section, a detailed algorithm is proposed as follows for concluding the personalized QoS evaluation method.

Algorithm PersonalizedQosEvaluation(S)

Require: Service: S, Service Records: *Records*, Service Request: *ServR*, threshold value: *tv*

```
1:  extract sR from ServR
2:  extract Records from historical log about S
3:  for all record R ∈ Records do
4:      extract aR from R
5:      aR.weight = computeLength(sR,aR)     //(1),(2)
6:      if aR.weight <= tv then
7:          fR.add(R)
8:      end if
9:  end for
10:  for all criteria q ∈ QoS do
11:     computeEq(fR);                       //(3)
12:     computeVq(fR);                       //(4)
13: end for
```

4 Performance Analysis and Optimization

In the following, the effectiveness and efficiency of our proposed method are both discussed, and an optimization method is devised to improve the efficiency of the method.

4.1 Effectiveness Analysis

The main purpose of our method is to evaluate each QoS criterion respectively for candidate services. Our method can be used before the overall quality evaluation of a service based on all these QoS criteria. Corresponding technique to compute overall quality has preference-oriented method [11], Simple Additive Weighting technique used in [5], and so on. Focusing on evaluation of single quality criterion, there are two methods for comparison, in which one is our method based on collaborative, and the other is the method used on [5] for computing quality criteria for elementary services. Take success rate for example, the value of the success rate is the ratio between the count of successful invocation and that of all history invocations without collaborative filtering in the later. As explained in Section 2 and experimental result showed in Fig.2, success rate is affected by both service provider and consumers. So our QoS evaluation method is more impartial for service provider and more accuracy to show service's quality to consumers. Even for some quality criteria that are not affected by service consumer, our method can still work on returning the same good value as former method as none historical QoS records is filtered.

The reputation is a special quality criterion that evaluates the trustworthiness of service providers. However, ensuring the veracity of reputation reports is a critical issue [6]. Focusing on above issue, one method is to compute the reputation by comparing the quality level that providers promise to the quality requirements. However, it neglects the consumers' subjective satisfactions for service usage, which is useful especially when consumer satisfaction for a single quality criterion is not linear proportional to the concordance between required QoS and delivered QoS. Our method can ensure the veracity of reputation reports by collaborative filtering malicious consumers' reputation based on similarity of service requests.

4.2 Efficiency Analysis and Optimization

Firstly, the candidate services set for personalized QoS evaluation can be reduced as small as possible by matchmaking with published QoS information. The matchmaking is based on the tenet that a service's actual quality cannot be better than the published quality. Non-skyline services are also not in the candidate services set. The count of candidate services is denoted as c. Then following considers one service for personalized QoS evaluation. Assuming the records number of execution log is r, the time complexity of collaborative filtering process is $O(r \times n)$, where n is the number of dimensions mentioned in formula(1) and (2). And the time complexity of predictive QoS computing is $O(m \times q)$, where m is size of left records and q is the size of QoS criterion. As $m \leqslant r$ and $n \leqslant q$, the overall time complexity of evaluating all candidate services is $O(c \times r \times q)$.

Usually, q is limited, and is a constant for a set of functionally equivalent services. However, r is an increasing variable over time with the growing number of service invoking. And r can be an instable variable that causing bad performance of our method. A straightforward optimization method is cutting r by a time threshold, in which old records are not considered in QoS evaluation. We devise another optimization method based on user clustering, detailed as follows:

Step1: Cluster user requests of historical invoking using K-Means algorithm.

Step2: Find the representative user request for each cluster produced in ***Step1***.

Step3: Evaluate the QoS of the representative user request for each cluster using our proposed method.

Step4: Response a service request, and compute out the nearest cluster of user request, followed returning the corresponding QoS evaluation result.

It is worth noting that ***Step1*** ～ ***Step3*** can be done offline periodically according the update of historical records, and ***Step4*** is done online for responding a new service request. Supposed the number of clusters is *k*, the time complexity of dynamic personalized QoS evaluating can be reduced to $O(k \times q)$, where k<<r.

The effectiveness and efficient of our method is verified by theoretical analysis. As a component of service selection system or service recommendation system, QoS evaluation cannot be experimented alone, our future work will be focus on applying our method into service selection for experimentation. In addition, our work has been in part ground on literature works in the area of SLA and QoS monitoring. SLA records a consumer's service requirement and QoS monitoring collects the historical execution QoS information.

5 Related Work

QoS is introduced into Web service in early literature [1], and [2], and the QoS criteria considered in these literatures contains availability, response time, throughput, security properties, accessibility, integrity, and regulatory and et al. QoS-aware service ranking, selection and composition get extensive research. Alrifai et al. [4] propose an efficient QoS-aware service composition method combining global optimization with local selection. Zeng et al. [5] described two service selection approaches, one based on local selection of services and the other based on global allocation of tasks to services using integer programming. Skoutas et al. [12] introduce service dominance scores based on multi-criteria dominance relationships for ranking and clustering Web services.

QoS computation or evaluation is a prelude of above work, and is concerned in literature [7], [11], [13] and so on. Rosario et al. [13] introduce soft probabilistic contract and use confidence interval and confidence to demonstrating service quality. Zheng et al. [7] evaluate real-world Web services by invoking these services in distributed manner and statistic QoS experience. However it is unrealistic for potential user to evaluate candidate services personally in this manner, and realistic method is estimate the QoS with historical invoking records. Liu et al. [11] extend QoS model with domain specific criteria and give a fair and open QoS computation method, in which all QoS information published by providers, from execution monitoring and requester's feedback are considered. However, it neglects the service consumer-side effect for execution QoS. Specially, trustworthiness or reputation of services is studied in literature [6], [14]. A method is developed for propagating reputation received by a composite service to its component services by Nepal el al [14]. Limam et al. [6] propose a reputation computation model based on automatic feedback computation for assessing software service quality and trustworthiness at selection time.

Similarly to our work, Thio et al. [15] introduce Client-Side Performance Estimation into Web service recommendation. The metrics related to client-side performance used in this paper are latency, transfer rate and throughput, which is similar to network environment in our proposed Service Request Model. Ivanovie et al. [16] address the issue of data-aware adaptation for service orchestrations. In our paper, a service request model is proposed for collaborative filtering and considers both network environment and input data, but also price, as higher price consumer pay, high quality provider provide. And a user-centric QoS evaluation method is devised for personalized service requests.

6 Conclusion

To handle the impartiality of QoS evaluation, a QoS evaluation method for personalized service requests is put forward in this paper. Specifically, a service request model is proposed to specify consumer-side affect for delivered QoS. Finally, the effectiveness of our method is validated, and an optimization method is proposed to improve the QoS evaluation efficiency. Our future work will focus on system implementing and experimental verification.

Acknowledgments. This paper is partly supported by the National Science Foundation of China under Grant No. 60721002, 61073032 and 60736015, and Jiangsu Provincial NSF Project under Grants BK2008017.

References

[1] Menasce, D. A.: QoS Issues in Web Services. IEEE Internet Computing, 6(6), 72-75 (2002).

[2] Mani, A., Nagarajan, A.: Understanding Quality of Service for Web Service. IBM Developer Works, https://www.ibm.com/developerworks/webservices/library/ws-quality.html (2002).

[3] Alonso, G., Casati , F., Kuno, H. A. , Machiraju, V.: Web Services: Concepts, Architectures and Applications, Springer, Heidelberg (2004).

[4] Alrifai, M., Risse, T.: Combining Global Optimization with Local Selection for Efficient QoS-aware Service Composition. In: 18th International Conference on World Wide Web, pp. 881-890 (2009).

[5] Zeng, L., Benatallah, B., Ngu, A. H. H., Dumas, M., Kalagnanam, J., Chang, H.: QoS-Aware Middleware for Web Services Composition. IEEE Trans on Software Engineering, 30(5), 311-327 (2004).

[6] Limam, N., Boutaba, R.: Assessing Software Service Quality and Trustworthiness at Selection Time. IEEE Trans on Software Engineering, 36(4), 559-574 (2010).

[7] Zheng, Z., Zhang, Y., Lyu, M. R.: Distributed QoS Evaluation for Real-World Web Services. In: 8th International Conference on Web Service, pp. 83-90 (2010).

[8] Brezillon, P., “Focusing on context in human-centered computing,” IEEE Intelligent Systems, 18(3), 62-66 (2003)

[9] Zukerman, I., Albrecht, D. W.: Predictive Statistical Models for User Modeling. User Model. User-Adapt. Interact. 11(1-2), 5-18 (2001).

[10] Su, X., Khoshgoftaar, T. M.: A Survey of Collaborative Filtering Techniques. Adv. Artificial Intellegence (2009).

[11] Liu, Y., Ngu, A. H.H., Zeng, L.: QoS Computation and Policing in Dynamic Web Service Selection. In: 13th Internation Conference on World Wide Web, pp. 66-73 (2004).

[12] Skoutas, D., Sacharidis, D., Simitsis, A., Sellis, T.: Ranking and Clustering Web Services Using Multicriteria Dominance Relationships. IEEE Trans on Service Computing. 3(3), 163-177 (2010).

[13] Rosario, S., Benveniste, A., Haar, S., Jard, C.: Probabilistic QoS and Soft Contracts for Transaction-Based Web Services Orchestrations. IEEE Trans on Service Computing. 1(4), 187-200 (2008).

[14] Nepal, S., Malik, Z., Bouguttaya, A.: Reputation Propagation in Composite Services. In: 7th International Conference on Web Service, pp. 295-302 (2009).

[15] Thio, N., Karunasekera, S.: Web Service Recommendation Based on Client-Side Performance Estimation. In: Proceedings of the 2007 Australian Software Engineering Conference pp. 81-89 (2007).

[16] Ivanovie, D., Carro, M., Hermenegildo, M.: Towards Data-Aware QoS-Driven Adaptation for Service Orchestrations. In: 8th International Conference on Web Service, pp. 107-114 (2010).