Performance Evaluation Of Service-Oriented Systems

Ansila Henderson, Soja Salim

MTech student, Department of Computer Science and Engineering, SCTCE, Pappanamcode, Trivandrum, India
Assistant Professor, Department of Computer Science and Engineering, SCTCE, Pappanamcode, Trivandrum,India.
Email: ansilahenderson@gmail.com

ABSTRACT: The exponential development of Web service results in high quality service-oriented systems an urgent and fundamental research problem. The Web services were developed by different organizations and put forward various functionalities and Quality of Service (QoS) values. The selection of a Web service, for each activity of the work flow, meeting the user's requirement is an important contest. The Quality of Service Management Framework Based on User Expectations assembles the expectations as well as ratings from the users of a service. It analyse the quality of the service only at the time a demand for the service is made and only using the ratings that have similar expectations. The collaborative filtering approach calculates QoS values of Web services. It makes Web service proposal by taking benefits of former usage events of service users. The Transactional and QoS- Aware selection algorithm deals with the concern of choosing and organizing Web services not only according to their functional requirements but also to their operational properties and QoS characteristics. The Web service QoS prediction framework, offers time-aware personalized QoS value prediction service for different service users. The online performance prediction framework predicts performance efficiently at run time.

Keywords : Service-Oriented Architecture (SOA); Quality of Service (QoS) ; Quality Assessment (QA); Service Providers (SP)

1 INTRODUCTION

WITH the exponential growth of Web services, service-oriented architecture (SOA) has become a major framework for building Web systems. In service computing, Web services put forward by different providers were learned and incorporated to realize complex tasks. A service-oriented system is composed of multiple Web services interacting with each other over the Internet in a random manner. The Web services were developed by different organizations and put forward various functionalities, transactional properties and Quality of Service (QoS) values. The selection of a Web service, for each activity of the work flow, meeting the user's requirement is still an important contest. So, how to build high-quality service-oriented system becomes an essential research problem. The user-side observed performance is employed to evaluate the qualities of Web services [2]. The designers of service oriented applications have to decide from a broad pool of functionally matching or similar Web services for creating applications. Web services may be deployed in remote servers and accessed by users through Internet connections. The qualities of service-oriented applications were greatly influenced by the quality of the invoked Web services. The five different techniques for the performance evaluation and enhancement of the service-oriented system were taken under consideration. The Quality of Service (QoS) management framework based on user expectations[10] collects the expectations as well as ratings from the users of a service. It calculates the quality of the service only at the time a request for the service is made and only using the ratings that have similar expectations. The collaborative filtering approach predicts QoS values of Web services. It makes Web service recommendation by taking advantages of past usage experiences of service users. The Transactional and QoS- Aware selection algorithm addresses the issue of selecting and composing Web services not only according to their functional requirements but also to their transactional properties and QoS characteristics. In this method, the designer focuses on the construction of the desired functionalities of the applications leaving the complex process of selection of Web Services based on their QoS and transactional behaviour to the composition manager. The Web service QoS prediction framework provides time-aware personalized QoS value prediction service for different service users. In order to learn the latent features of users, services, and time, this employs tensor factorization technique to fit a factor model to the user-service-time tensor. This framework predicts missing QoS values based on the past QoS experience and the available QoS information in the current time interval. If no QoS information is available in the current time interval, it purely depends on the past experience. The online performance prediction framework provides personalized service-oriented system performance prediction efficiently at run time. Using the past Web service usage experience from different users, it builds feature models and employs time series analysis techniques on feature trends to make personalized performance prediction for different service users. Among the five different approaches the online performance prediction framework outperforms the state-of-the-art performance prediction approaches in terms of prediction accuracy.

2 PERFORMANCE EVALUATION OF SERVICE-ORIENTED SYSTEMS

The five different techniques for the performance evaluation and enhancement of the service-oriented system are as follows.

2.1 QUALITY OF SERVICE MANAGEMENT FRAMEWORK

Several service providers (SP) may put forward same services but with diverse qualities in an SOC environment. An agent should pick a service that encounters the required capacity with the lowest possible price as well as the quality requirement. A variety of schemes for developing, calculating and observing QoS were proposed in the literature [3], [4], [5], [6], especially for web services and multi-agent systems [7], [8], [9]. A familiar methodology is to gather quality ratings from the users of a service and then combine them in order to obtain the quality of the service. The quality of service management framework is based on user expectations [10]. Different users may have different expectations on the quality of a service and their ratings liable to be closely interrelated to how their expectations were met. Collecting quality ratings only from the users was not enough for getting a consistent or exact quality meas-

ure for a service. So this framework collects expectations as well as ratings from the users of a service. It determines the quality of the service only at the time a request for the service has made and only by means of the ratings that have similar expectations. The QoS ratings collected from the users would be used in the calculation of QoS by the Quality Assessment (QA). The QA agent consists of two major components-the Rating Collector and the Rating Calculator. The Rating Collector requests for the quality ratings from the users. The Rating Calculator calculates QoS from the collected ratings. To combine individual ratings into a global review of quality for a given service, two calculations were necessary [10]:

1. Combine individual ratings for each attribute in the given service into an aggregate rating.
2. Combine the ratings for individual attributes into an overall rating for the service.

This method has certain limitations. First, it must consider how user expectation would be best represented. The real-number based representation was limited that does not allow the user to specify the range of expectations. Second, better techniques have to be developed for matching expectations, precisely for expectations that were not stated as simple real numbers. Also, it is necessary to deal with what happens when there are no matching expectations when assessing QoS, or the so-called "cold start" problem [11].

2.2 Collaborative Filtering

The concept of user-collaboration for the Web service QoS information sharing between service users was proposed by the inspiration of success of YouTube and Wikipedia. Instead of contributing videos (YouTube) or knowledge (Wikipedia), the service users were encouraged to contribute their individually observed past Web service QoS data [12]. In the collaborative filtering method a prototype called Web Service Recommondation (WSRec) was implemented and deployed to the Internet for QoS Prediction. This method doesn't require real-world Web service invocations in order to predict user-dependent QoS values. The procedure for user collaborative QoS data collection mechanism was described as follows:

1. A service user supplies past Web service QoS data to the centralized server WSRec. The service users who demand QoS value prediction services were named as active users.
2. WSRec chooses similar users from the training users for the active user. Training users correspond to the service users whose QoS values were stored in the WSRec server and utilized for making value predictions for the active users.
3. WSRec forecasts QoS values of Web services for the active user.
4. WSRec makes Web service suggestions based on the predicted QoS values of different Web services.
5. The service user collects the predicted QoS values as well as the results, which can be employed to support decision making.

In the user-collective mechanism, the active user who adds more Web service QoS data would get more accurate QoS value predictions. Thus, the service users were encouraged to contribute their past Web service QoS data. The similarities of different service users and different Web services were computed using the Pearson Correlation Coefficient. Pearson Correlation Coefficient (PCC) can be easily implemented and have high accuracy. So it was used in a number of systems for similarity computation. But the Pearson Correlation Coefficient (PCC) would overestimate the similarities of service users who were actually not similar but happen to have similar QoS experience on a few co-invoked Web services. So a significant weight was employed to lower the impact of a small number of similar co-invoked items. Predicting missing values for the user-item matrix would improve the prediction accuracy of active users [13]. Subsequently, a missing value prediction approach was proposed for making the matrix denser. The value would be predicted by employing similar users or items of a missing value in the user-item matrix. Traditional Top-K algorithms rank the neighbours based on their PCC similarities [12] and select the top k most similar neighbours for making missing value prediction.

2.3 Transactional And QoS-Aware Selection

The conventional QoS-aware composition methods do not care about the transactional restrictions during the composition process. The Transactional and QoS-aware Selection (TQoS) deals with the matter of selecting and composing Web services according to their functional requirements, their Transactional Properties (TPs) and QoS characteristics. The selection algorithm that suits user's preferences were prompted as weights over QoS criteria and as risk levels defining semantically the transactional requirements. Each selected component Web Service of a composite one would be locally the best QoS Web Services among all the Web Services fulfilling the global transaction requirement. The user has to define only a global transaction requirement and no need to define the possible termination states of all component Web Services. When some functionally equivalent Web Services were available to do the same activity, their QoS properties such as price, reliability, availability, and reputation would be important in the selection process. A model was employed to acquire the descriptions of these properties from a user viewpoint. Such models must take into account the fact that QoS involves multiple dimensions [14]. The input of the TQoS-driven selection algorithm would be a workflow (WF) composed of n activities and the output is a Transactional Composite Web Service (TCWS) corresponding to a list of elementary Web Services or Composite Web Services allocated to each activity of the input workflow. When a Web Service was allocated to an activity of the workflow, its transactional property influences the selection of the Web Service for the next activities.

2.4 Time-Aware Personalized QoS Prediction Framework

Time-Aware Personalized QoS Prediction Framework [15] collects time-aware QoS information from geographically distributed service users. It combines the local information to get a global user-service-time tensor using tensor factorisation. User-specific, service-specific and time-specific latent features were extracted from the past QoS experiences of different service users by using tensor factorization. The unknown QoS values were estimated by analyzing how the user features were applied to the consequent service features and time features. To automatically record QoS values of invocations and to add the local records to the server, a user-side light-weight middleware was used for the service users. Thus more invocation results could be obtained from other service users. Within

the middleware, there were three management components:
1. Monitor,
2. Collector and
3. Predictor

Monitor was responsible for monitoring the QoS performance of Web services. When users send invocations, it would monitor the QoS performance. Collector would supply local QoS information to other users and collect shared QoS information from other users. Predictor provide time-aware personalized QoS value prediction based on local and other users' QoS information collected by Collector [15]. Web services can update automatically with better ones at run-time. It requires time-specific QoS performance of Web services. The efficient tensor factorization technique was used to fit a factor model to the user-service-time tensor. To make additional missing entries prediction the factorized user-specific, service-specific and time-specific matrices were utilized. Thus it develops a high-quality low-dimensional feature representation of users, services and time by analyzing the user-service-time tensor.

2.5 Online Performance Prediction Framework

The Online Performance Prediction Framework [1] extends Time-Aware Personalized QoS Prediction Framework to control updated QoS information at run-time. It performs time series analysis for prediction. It collects time aware performance information from geographically distributed service users. A service user usually invokes only a small number of Web services in the past. It would observe performance of only the invoked Web services. So the collected performance information usually becomes sparse. This framework uses a set of latent features to characterize the status of Web services and users. Latent features were orthogonal representation of the decomposed results of physical factors. From the collected service performance information the latent features of users and services in the past time slice were extracted. Thus it estimates the features of users and services in the current time by analyzing the trend of the feature changes. The online performance prediction mechanism contains four phases:

1. Each service user maintains local performance records of the Web services.
2. Local Web service usage experiences were uploaded to the performance centre. Each user is encouraged to contribute its local records to obtain performance prediction service from the performance centre. A service user can obtain more accurate performance prediction results from the performance centre by contributing more individually observed Web service performance records.
3. A performance model was built in the performance centre for personalized service performance prediction. For that it performs time series analysis on the extracted time-specific user features and service features.
4. The overall performance of a service oriented system was predicted based on the analysis of service compositional structures from the service level performance information. When the most recent service performance information become available, an online prediction algorithm was applied. It would quickly update the performance model. In order to catch the performance trend it doesn't require any effort of recalculation.

The Online Performance Prediction framework uses a matrix factorization technique to fit a feature model to user-service matrix in each time slice. The factorized user-specific and service-specific features are utilized to make further performance prediction. The idea behind the feature model is to derive a high-quality low-dimensional feature representation of users and services by analyzing the user-service matrices. The service performance prediction was conducted in two phases: offline phase and online phase [1]. In the offline phase, the trends of user features and service features were modelled by using the performance information collected from all the service users. The features of users and services in the next time slice were calculated based on the evolutionary algorithm. The predicted features were further applied for calculating the predicted performance of services in the next time slice. In the online phase, the newly observed service performance information by users at runtime was integrated into the feature model build in the offline phase. The feature model was updated efficiently by using the incremental calculation algorithm. Thus it captures the latest trend in ensuring the prediction accuracy. In online phase, an incremental algorithm was used for efficiently updating the feature model built in offline phase at runtime since new performance data were collected in each time slice. The overall performance of a system consists of service response time and local execution time. Local execution time refers to the computation time between service invocations in local system. Since the variance of system performance at runtime is mainly due to the highly varying service response time, local execution time, which is relatively constant at runtime, is not included in the defined system level performance. With the aggregation approach, designers of service-oriented systems could estimate the performance of systems at design-time. At runtime, the user observed system level performance could be predicted automatically. Once the system performance has decreased at runtime, bad performance services could be quickly identified by analyzing the system structure in a top down way. To improve the system performance, dynamic service composition techniques were employed by replacing the long response time services with better one.

3 Comparison

1. In QoS management framework, the groups of users who have similar expectations were formed dynamically, based on the given QoS assessment request and the criteria for similarity measures. But how user expectation may be best represented was not considered. The quality perceived by the user could play an important role in deriving more accurate quality assessment. But it was not used in quality calculation.
2. In collaborative filtering, users were categorised into several groups and the users in the same group were considered to have the same expectations. This common expectation was then used to determine or predict if a given item would be of any interest to a particular user.
3. The TQoS, properly selects component Web Services in order to obtain a Transactional Composite Web Service. It maximizes the user satisfaction in terms of QoS criterion. It also satisfies the transactional requirements set by the user and by the input workflow. The designer could focuse only on the construction of the desired functionalities of the applications. The TQoS algorithm was scalable because the users need to define only a

global transaction requirement.

4. Time-Aware Personalized QoS Prediction Framework was based on tensor factorization prediction method. This framework predicts missing QoS values based on the past QoS experience and the available QoS information in the current time interval. If no QoS information was available in the current time interval, it purely depends on the past experience.
5. The online prediction algorithm was used to perform time series analysis for prediction. This framework could precisely predict service performance.

Since online prediction framework could precisely predict service performance, the service-oriented system will updated by integrating potentially optimal services at runtime. The system performance of static composition was unstable at runtime. This is because the performance of some selected services become unstable, which impacts the system overall performance. The system performance of dynamical composition maintains stable in a good level, which indicates the effectiveness of this framework.

4 CONCLUSIONS

In service computing, Web services offered by different providers integrated to implement complicated functions. The service-oriented system consists of multiple Web services interacting with each other over the Internet in an arbitrary way. So, how to build high-quality service-oriented systems becomes an urgent and crucial research problem. The purpose of performance evaluation was to monitor the current system performance status and allow designers to make adjustments in order to guarantee the performance in the future. This requires frequent performance evaluation. The Online performance prediction framework estimates the user observed performance of service-oriented systems by employing the past usage experiences of different users. This framework efficiently predicts the performance of service-oriented systems online. Using the past Web service usage experience from different users, it builds feature models and employs time series analysis techniques on feature trends to make personalized performance prediction for different service users. Among the five different approaches the Online Performance prediction framework outperforms the state-of-the-art performance prediction approaches in terms of prediction accuracy. More techniques must investigate to improve the prediction accuracy (e.g., data smoothing, utilizing content-aware information, and so on).

REFERENCES

[1] Z. Z. Yilei Zhang and M. R. Lyu, “An online performance prediction framework for service-oriented systems,” IEEE Trans.Sys. man and Cyber., 2014.

[2] Y. Z. Z. Zheng and M. R. Lyu, “Investigating qos of real-world web services,” IEEE Trans.Serv. Comput.

[3] C. A. H. L. Aurrecoechea, C., “A survey of qos architectures,” Multimedia Systems Journal,vol. 6, pp. 138–151, 1998.

[4] S. M. Chalmers, D., “A survey of quality of service in mobile computing environments,”IEEE Communications Surveys and Tutorials 2, pp. 2–10, 1999.

[5] S. D. K. B. W. R. Lee, Y.W., “Aimq:a methodology for information quality assessment,”Information and Management, vol. 40, p. 133–146, 2002.

[6] Z. V. B. L. Parasuraman, A., “Reassessment of expectations as a comparison standard in measuring service quality: implications for future research,” Journal of Marketing,vol. 58, pp. 201–230, 1994.

[7] H. D. Trzec, K., “Intelligent agents for qos management,” Proceedings of First InternationalConference on Autonomous Agents and Multi Agent Systems, pp. 1405–1412, 2002.

[8] S. M. Yu, B., “An evidential model of distributed reputation management,” Proceedings of First International Conference on Autonomous Agents and MultiAgent Systems, pp. 294–301, 2002.

[9] B. B. D. M. K. J. Zeng, L., “Quality driven web service composition,” Proceedings of Twelfth International Conference on World Wide Web, pp. 411–421, 2003.

[10] W. A. G. ikas Deora, J. Shao and N. J. Fiddian, “A quality of service management framework based on user expectations,” Springer-Verlag Berlin Heidelberg, vol. LNCS 2910,p. 104–114, 2003.

[11] P. A. U. L. P. D. Schein, A., “Methods and metrics for cold-start recommendations,”Proceedings of 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 253–260, 2002.

[12] M. R. L. Zibin Zheng, Hao Ma and I. King, “Qos-aware web service recommendation by collaborative filtering,” IEEE Trans. Serv. Comput, vol. 4, 2011.

[13] X. Z. X. Su, T.M. Khoshgoftaar and R. Greiner, “Imputation-boosted collaborative filtering using machine learning classifiers,” Proc. ACM Symp. Applied Computing, pp. 949–950, 2008.

[14] M. M. J. El Haddad and M. Rukoz, “Tqos: Transactional and qos-aware selection algorithm for automatic web service composition,” IEEE Transactions on Services Computing, pp. 73–85, 2010.

[15] Z. Z. Y. Zhang and M. Lyu, “Wspred: A time-aware personalized qos prediction framework for web services,” Proc. ISSRE, pp. 210–219, 2011.