International Conference & Workshop on Recent Trends in Technology, (TCET) 2012
Proceedings published in International Journal of Computer Applications® (IJCA)

An Approach to Web Services Configuration based on QoS

Yogini Marathe
Department of Computer Engineering, Pune Institute of Computer Technology
Pune, India

Pravin Game
Department of Computer Engineering, Pune Institute of Computer Technology
Pune, India

ABSTRACT

In a Web environment, Web services will typically be configured together to serve a specific purpose. Many web services provide similar functionality however they have different nonfunctional properties (e.g. price). A lot of research is being made on how to configure, discover, and compose web services on the basis of their non functional characteristics. This paper aims at studying approaches for QoS based configuration of Web Services, analyze the advantages and drawbacks of the various approaches and later we propose an advancement needed in the approach in order to make the QoS aware configuration more effective in the real time environment.

General Terms

Web Services

Keywords

Quality of Service, Service selection, Web Services

1. INTRODUCTION

In today's world, many Enterprises are adopting Service-Oriented Architecture for business processes within the organization and Web Services are the obvious choice for SOA implementation. Web service providers publish their web services in common registry called Universal Description, Discovery and Integration (UDDI). Then, application programs search the registry for required web services. These registered web services are combined together to form a configuration in order to serve all the needs of the end users. Run time performance of these composite web services is of utmost importance for distributed applications. While choosing the Web services for configuration, it is important to evaluate functional as well as non functional characteristics. Functional aspects include, the functionality provided by the service and under what circumstances the functionality is provided. Non functional characteristics include quality of service (QoS). It is always possible that there are multiple services that can meet functional requirements however have different QoS attributes.

Norbert Bieberstein et al. have identified various non functional requirements that should be satisfied by the SOA based systems in [16]. They classify the non functional requirements as business constraints, technology constraints, runtime qualities and non runtime qualities. Our scope of QoS includes the runtime qualities such as performance, reliability, scalability, capacity, robustness, exception handling, accuracy, integrity, accessibility, availability, interoperability, security and network related QoS requirements [1]. These properties are applicable to both stand-alone Web services and Web services compositions [2]. Several researches are being done and approaches have been suggested to make use of QoS parameters while choosing web services.

The rest of the paper is organized as follows: Highlights of the similar work done are given in Section 2. An enhanced configuration approach is presented in Section 3. Results based on test data are shown in Section 4. Section 5 lists the conclusion and future work.

2. RELATED WORK

All S.Ran has proposed UDDI extension model in order to make it possible to discover the web services based on QoS attributes [3]. Middleware based approaches have been presented for creating web services compositions as well as for managing dynamically changing QoS requirements[2],[4]. Z. Zheng et al. have proposed collaborative filtering approach to predict QoS values of the web services [6]. C.Zhou et al. designed the QoS ontology to help in services discovery with required QoS as well as for measuring the QoS [7]. More and more research is being done on how to dynamically configure web services to meet user's non functional needs. To address this need, P. C. Xiong et al.have proposed a service function dependency configuration net based on Petri nets. Later an algorithm is proposed for choosing the best configuration that has the highest quality of service [5].

Work done by Xiong et al. [5] is being analyzed, appreciated and used as a reference by many others. Yang Huaizhou, Li Zengzhi highlight that the work done in [5] is good for modeling of service dependent relationship. However it is difficult to integrate the solution with current service orchestration engines because Orchestration process is not considered in the models. Two composite services, which are comprised of same component services, will differ in terms of QoS when their orchestration processes are different. To do appropriate evaluation whether the user's QoS requirements are satisfied or not orchestration process needs to be considered. To address the need, the service dependent relationships and the possible orchestration processes are synthetically reflected by an extended hierarchical Petri nets. Also a series of QoS formulas are presented [8].

George Zheng and Athman Bouguettaya propose web service mining framework that enables proactive discovery of interesting and useful service compositions. Usefulness and interestingness parameters are considered. Weighted function is used to compute overall Quality of the service. E.g. reliability attribute contributes positively whereas response time attribute contributes negatively while calculating QoS. They also use skyline approach for identifying interestingness of a particular service composition [9].

Authors of [5] themselves have improved the algorithm to consider multiple QoS attributes. Simple additive weighting is used to find overall QoS value of the configuration [10]. W.

Tan et al. have used colored Petri nets to capture both control and data aspects of business processes as well as services. Based on data relations and composition rules a method is proposed to derive all the possible composition candidates given a service portfolio. [11] tackles the problem of service composition using Petri nets decomposition.

D. Bruneo et al. present a novel technique that, starting from the WS-BPEL statements and assuming that the non-functional parameters of the services involved in the process are known, investigates the QoS of composed services. This is done by mapping the WS-BPEL process into non-Markovian stochastic Petri nets (NMSPNs). Both synchronous and asynchronous WS-BPEL invocation of external Web services is taken into account. The main advantage of the proposed technique is to implement a stochastic analysis of the evaluated quantities so as to make both transient and steady state evaluations. This helps in evaluating several different conditions ranging from the worst to the best case, and not only the average case as provided by the other existing techniques [12].

Xitong Li et al. proposed to model web services using colored Petri nets. Web services are modeled as SWN (Service Workflow Net) based on colored Petri nets. It shows observable data flow between the component web service. As stated in [5], due to dynamic environment, a web service involved in a composition may fail to respond to client's requests or to maintain satisfying QoS level. In any case, an alternative Web Service of similar behavior needs to be identified and used as a substitute for failed service. In this paper, authors provide sufficient conditions of context independent similarity and corresponding algorithm to verify similarity. Also a tool is developed to automate the verification of similarity between web services using this algorithm [13].

3. APPROACH

In addition to the various approaches suggested above, we suggest an enhancement to use historical data of the composite services. This should help to find out a configuration which is optimal as well as has been consistently giving the better performance in terms of user requested QoS parameters. Historical data of various QoS parameters such as response time, availability of the composition can be calculated as per the approaches suggested in [15]. There is a possibility that we can construct multiple optimal configurations from the available service candidates. So we should find out all possible optimal configurations as a first step. From this set of configurations, select the configuration that has consistently given better performance in the past. Advantage of selecting configuration based on past performance data is that we can guarantee the given configuration will meet user's requirements at runtime. All the approaches mentioned in section 2 use some formulae to calculate the Composite QoS value based on the values of atomic candidate services. However when these atomic services are combined together, the flow logic, calling sequences would introduce the delay, thus actual QoS achieved might be different than the calculated QoS. So it is important to consider the past performance of the configuration though theoretically the configuration is optimal.

3.1 Modeling

For modeling of the web service compositions, Petri nets and Directed Acyclic Graphs have been used widely [5], [14]. As shown in one of the approaches in [14] we use directed acyclic graph (DAG) for modeling the web services configuration. Following are various components in the system:

- $C = \{S_1, \ldots, S_n\}$ Composite service flow made up of individual service functions
- S_i = Service function class representing collection of services with same functional but different QoS properties; $Si \in C$
- S_{ij} = Service j providing functionality for service function class i
- $Q_{ij} = \{q_{ij}^1, \ldots, q_{ij}^x\}$ is a QoS vector for service S_{ij} ;The values can be average or worst case.
- $\Psi = \{\Psi_1, \ldots \Psi_m\}$ is expected QoS vector for configuration
- $G = \{V,E\}$ is a service candidate graph generated during processing where V is set of vertices representing all individual services and E is set of edges representing service interactions
- $H = \{C_i, \Psi_i\}$ is past runtime QoS data for configurations
- R = Optimal configuration suggested = $\{ S_1, \ldots, S_k \mid S_k \in S_{ij}\}$
- F = utility function that computes weighted QoS value for each service candidate

3.2 Utility Function Definition

Users would typically have multiple QoS objectives such as minimizing the response time while maximizing the throughput. All such requirement should be met while choosing the services. We will be using simple additive weighting while calculating the composite QoS value for each service. As described in [14] the utility function can be defined as below:

Suppose there are x QoS parameters to be maximized and y QoS parameters to be minimized then

$$F = \sum_{\alpha=1}^{x} \omega^{\alpha} * ((q_{ij}^{\alpha} - \mu^{\alpha})/\sigma^{\alpha}) + \sum_{\beta=1}^{y} \omega^{\beta} * (1 - ((q_{ij}^{\beta} - \mu^{\beta})/\sigma^{\beta}))$$

Where ω^{α} and ω^{β} are weights for each QoS attribute ($0 < \omega^{\alpha}$, $\omega^{\beta} < 1$) and $\sum_{\alpha=1}^{x} \omega^{\alpha} + \sum_{\beta=1}^{y} \omega^{\beta} = 1$. μ and σ are average and standard deviation values respectively for all candidates in a service class.

Table 1. Service Candidate List

Service Function	Candidates	Response Time	Availability
S1	S11	1	0.9
S1	S12	2	0.8
S2	S21	1.25	0.7
S2	S22	2	0.88
S3	S31	2.5	0.76
S3	S32	3	0.85
S3	S33	3	0.9
S4	S41	1.6	0.75
S4	S42	1.5	0.7
S5	S51	1	0.8
S5	S52	2	0.7
S5	S53	3	0.6

3.3 Algorithm

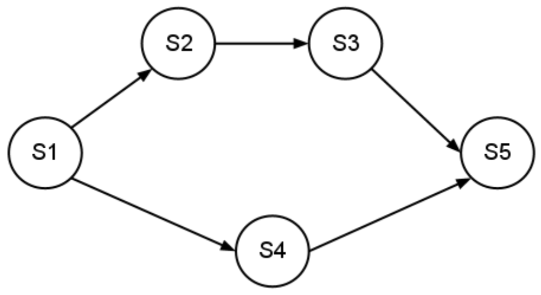

Fig 1: Service Flow Information

1. Inputs are the service functional flow information C as shown in Fig.1 and Service candidates list S_{ij} is as shown in Table 1.
2. From C and S_{ij} create Service candidate graph G (shown in Fig. 2) as below:
   a. $\forall$ Sij add a node v in G
   b. Add a link from every node S_{ij} to S_{jj} if there was a link from Si to Sj in C
   c. Add a dummy Start node S_s and End Node S_e whose QoS value is 0.
   d. $\forall$ v $\in$ V, add weight w to the incoming edges as value obtained by applying utility function for corresponding S_{ij}
3. Traverse the graph from start node to end node such that $\sum wj$ is maximum and QoS value of path meets the user defined QoS criteria. Function for graph traversal in explained in next section.
4. From the list of paths above, select the path such that historical QoS value Ψ is maximum

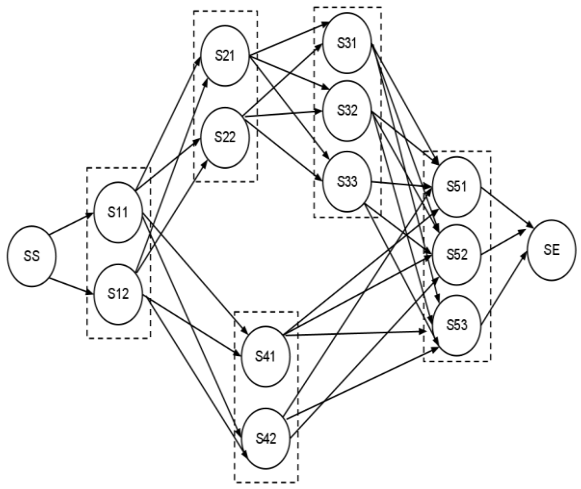

Fig 2: Service Candidate Graph

3.4 Graph Traversal Function Pseudo code

This function is based on shortest path finding algorithm, however instead of striving for minimum distance, the algorithm looks for nodes such that value of utility function is maximized [14].

```
startNode = Ss
nodeStack = Ss
pathsLists[0][0] = φ
while nodeStack !empty do
   X = adj(Ss)
   for all x ∈ X do
      if w = maximum or w = 0 then
         if Composite ofthepath _ expected  then
            push x
            Add Ss as previous node in x
         end if
      end if
      Ss = pop nodeStack
   end for
end while
pathLists = GetAllMarkedPaths()
```

GetAllMarkedPaths will traverse the graph from end node in backward direction and with the help of previous node information added at each traversed node, get all the paths.

4. EVALUATION EXAMPLE

We checked the correctness of the algorithm by considering two QoS parameters, Response time and Availability. Response time is the parameter that needs to be minimized whereas Availability is the parameter that should be maximized. As per the test data in Table 1, and weights used as 0.5 and 0.5, Table 2 shows the values for utility function of each service candidate.

Table 2. Utility values for Services

Service Name	Utility Value
S11	1.207
S12	-0.207
S21	0.5
S22	0.5
S31	0.537
S32	0.305
S33	0.657
S41	0.5
S42	0.5
S51	1.5
S52	0.5
S53	-0.5

Let expected QoS values of the configuration be Response time <= 7 ms. and service should be available at least 70 percent of the time. Using the utility values in table 2, algorithm produces following 4 different optimal paths out of 48 feasible paths:

1. $S_s \rightarrow S_{11} \rightarrow S_{21} \rightarrow S_{33} \rightarrow S_{51} \rightarrow S_e$
2. $S_s \rightarrow S_{11} \rightarrow S_{22} \rightarrow S_{33} \rightarrow S_{51} \rightarrow S_e$
3. $S_s \rightarrow S_{11} \rightarrow S_{41} \rightarrow S_{51} \rightarrow S_e$
4. $S_s \rightarrow S_{11} \rightarrow S_{42} \rightarrow S_{51} \rightarrow S_e$

Based on the historical data, suppose path 2 has yielded the best results in past, the output will be service configuration

$R = \{S_{11}; S_{22}; S_{33}; S_{51}\}$

5. CONCLUSION AND FUTURE WORK

Several approaches have been specified earlier for optimal QoS based compositions or configurations of web services. Our approach is an add-on to these approaches where we use information about historical behavior of entire composition for creating optimal service configurations. Advantage of selecting the configuration based on historical data is that the system will be more robust than the system if the configuration was selected as first available choice. The proposed approach is well suited for service compositions with sequential flow, in future we plan to work on other possible composition structure like parallel, conditional, loop structures.

6. REFERENCES

[1] QoS for Web Services: Requirements and Possible Approaches: http://www.w3c.or.kr/kr-office/TR/2003/ws-qos

[2] L.Z. Zeng, B. Benatallah, A.H.H. Ngu, M. Dumas, J. Kalagnanam and H. Chang, ”QoS-aware middleware for Web services composition”, IEEE Transactions on Software Engineering, Vol. 30, Iss. 5, May 2004, pp. 311-327.

[3] S. Ran, ”A Model for Web Services Discovery with QoS”, SIGecom Exchanges, vol. 4, no. 1, pp. 110, Mar. 2003.

[4] E. Wohlstadter, S. Tai, T. Mikalsen, I. Rouvellou, and P. Devanbu, ”GlueQoS: Middleware to sweeten quality-of-service policy interactions”, in Proc. 26th Int. Conf. Softw. Eng., Edinburgh, U.K., May 2004, pp. 189-199.

[5] P. C. Xiong, Y. S. Fan, and M. C. Zhou,”QoS-aware Web service configuration”, IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 38, no. 4, pp. 888-895, Jul. 2008.

[6] Zibin Zheng; Hao Ma; Lyu, M.R.; King, I,”QoS-Aware Web Service Recommendation by Collaborative Filtering”,IEEE Transactions on Services Computing, Vol 4. Issue 2, pp 140-152Feb 2011.

[7] Chen Zhou; Liang-Tien Chia; Bu-Sung Lee,”Semantics in service discovery and QoS measurement”,IT Professional, Vol 7. Issue 2, pp 29 - 34, Mar-Apr 2005.

[8] Yang Huaizhou, Li Zengzhi,”Improving QoS of Web Service Composition by Dynamic Configuration”,Information Technology Journal, Vol.9 Issue.3, pp.422, 2010.

[9] G. Zheng, A. Bouguettaya,”Service mining on the web”,IEEE Transactions on Services Computing, Vol 2, Issue 1, Jan-Mar 2009.

[10] P. C. Xiong, Y. S. Fan, and M. C. Zhou,”Web Service Configuration Under Multiple Quality-of-Service Attributes”,IEEE Transactions on Automation Science and Engineering, Vol. 6 Issue 2, Apr 2009.

[11] W. Tan, Y. Fan, M. Zhou, and Z. Tian,”Data-Driven Service Composition in Enterprise SOA Solutions: A Petri Net Approach”,IEEE Transactions on Automation Science and engineering, Vol. 7, No. 3, Jul 2010

[12] D. Bruneo, S. Distefano, F. Longo, M. Scarpa,”QoS Assessment of WS-BPEL Processes through non-Markovian Stochastic Petri Nets”, IEEE, 2010

[13] X. Li, Y. Fan, Q. Z. Sheng, Z. Maamar, and H. Zhu,”A Petri Net Approach to Analyzing Behavioral Compatibility and Similarity of Web Services”,IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, Vol. 41 Issue 3, May 2011

[14] Yu, Tao and Zhang, Yue and Lin, Kwei-Jay,”Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints”,ACM Trans. Web 1 1, Article 6, May 2007.

[15] Fauvet, Marie-Christine and Dumas, Marlon and Benatallah, Boualem,”Collecting and Querying Distributed Traces of Composite Service Executions”, Confederated International Conferences DOA, CoopIS and ODBASE 2002, Robert Meersman and Zahir Tari (Eds.). Springer-Verlag, London, UK, UK, 373-390.

[16] Norbert Bieberstein, Sanjay Bose, Marc Fiammante, Keith Jones, Rawn Shah,”Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap”,IBM Press Publication, 2005.