ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

IARJSET

International Advanced Research Journal in Science, Engineering and Technology

ICRAESIT - 2015

B V Raju Institute of Technology, Narsapur

Vol. 2, Special Issue 2, December 2015

A Model for Automated Web Service Composition in Dynamic Environment

Dr. Neelakantappa. M [1], Dr. Amjan Shaik [2], M. Revathi [3]

Professor of IT, BVRIT, Narsapur, Medak (District), India[1, 2]

M.Tech (SE), BVRIT, Narsapur, Medak (District), India[3]

Abstract: The ability to perform web service discovery and composition automatically and dynamically is essential and has emerged as an important research topic. Automated web service composition deals with the significant increase in the number of available services over time, as well as frequent changes in their definitions. It enables significantly faster responses to user queries for composite services, compared to the manual case. Also, it produces compositions up-to-date with the latest web service definitions, despite the dynamic environment. Also permits approximate composition and enables the quality assessment of the produced composite services in terms of accuracy by using AI techniques especially planning. The framework maintains compatibility with the current standards, to ensure interoperability, and it is independent from specific planners. The model devised in this paper focuses on the addition of the OWL-S descriptions of produced composite services in the registry of available services, to explore the possibility to accelerate the composition process.

Keywords: MMKP, MILP, LP, BPEL.

I. INTRODUCTION

Numerous procedures like accounting, eScience, finances, multimedia programs and supply chain management are shifting, regarding the ad-hoc method for service composition. The composition concern gets most difficult with large amount of services obtainable that are growing each day and provide the same performance with variations of QoS. The selection of component services from a set of offered services usually results in complicated decision concern. The purpose of a service selection algorithm is mapping of every single process chosen to performance to an applicable service from a set of obtainable services. This permits an optimized QOS in regards to the whole process and the user's specifications.

In web services of compelling nature, the QoS assessments have concerns of deviations. This necessitates during execution decisions for selecting suitable algorithms as well as dynamic replacement of services in real-time (e.g. in multimedia programs) with no compromising the effectiveness. A composition consult (e.g. in a workflow language like BPEL [1]) may be patterned as Multi Choice Multidimensional Knapsack (MMKP) issue that even so is an extremely difficult concern [2]. For an optimized QOS in regards to the overall process and the user's specifications to MMKP incurs massive cost. The MILP (Mixed Integer Linear programming techniques) [3], have concerns of inadequate scalability as well as recent strategies [4, 5] cannot effectively maintain run-time specifications. In this document an efficient and scalable heuristic technique for QoS-based service selection is evaluated with the following steps,

1. The complete QoS optimization concern is portioned into many sub-problems by native QoS optimization for complete effective solution.
2. The concern decomposition outcomes as explained in this document are applicable to a distributed design containing a service assembler also a group of distributed service agents.

II. SYSTEM MODEL

In our model we assume that we have a universe of web services S which is defined as a union of abstract service classes. Each abstract service class $S_j \in S$ (e.g. flight booking services) is used to describe a set of functionally-equivalent web services (e.g. Lufthansa and Qantas flight booking web services). In this paper we assume that information about service classes is managed by a set of service brokers as described. Web services can join and leave service classes at any time by means of a subscription mechanism. We also distinguish between the following two concepts:

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

IARJSET

International Advanced Research Journal in Science, Engineering and Technology

ICRAESIT - 2015

B V Raju Institute of Technology, Narsapur

Vol. 2, Special Issue 2, December 2015

- An abstract composite service, which can be defined as an abstract representation of a composition request $CS_{abstract} = \{S_1, \ldots, S_n\}$. $CS_{abstract}$ Refers to the required service classes (e.g. flight booking) without referring to any concrete web service (e.g. Qantas flight booking web Service).
- A concrete composite service, which can be defined as an instantiation of an abstract composite service. This can be obtained by bounding each abstract service class in $CS_{abstract}$ to a concrete web service s_j, such that $s_j \in S_j$.

QoS VECTOR

In our study we consider Quantative non-functional properties of web services, which can be used to describe the quality of a service s. We use the vector $Q_s = \{q_1, q_2, \ldots, q_r\}$ to represent these properties. These can include generic QoS attributes like response time, availability, price, reputation etc, as well as domain-specific QoS attributes like bandwidth, video quality for multimedia web services. The values of these QoS attributes can be either collected from service providers directly (e.g. price), recorded from previous execution monitoring (e.g. response time) or from user feedbacks (e.g. reputation).

The set of QoS attributes can be divided into two subsets: positive and negative QoS attributes. The values of positive attributes need to be maximized (e.g. throughput and availability), whereas the values of negative attributes need to be minimized (e.g. price and response time). For the sake of simplicity, in this paper we consider only negative attributes (positive attributes can be easily transformed into negative attributes by multiplying their values by -1). We use the function $q_i(s)$ to determine the i-th quality parameter of service s. The QoS information of web services from class S is managed by the responsible service broker of this class.

QoS COMPUTATION OF COMPOSITE SERVICES

The QoS value of a composite service is decided by the QoS values of its component services as well as the composition model used (e.g. sequential, parallel, conditional and/or loops). In this paper, we focus on the service selection algorithm for QoS-based service composition, and its performance on the sequential composition model. Other models may be reduced or transformed to the sequential model. Techniques for handling multiple execution paths and unfolding loops from can be used for this purpose.

The QoS vector for a composite service CS is defined as $Q_{CS} = \{q'_1(CS), \ldots, q'_r(CS)\}$ where $q'_1(CS)$ represents the estimated QoS values of a composite service CS and can be aggregated from the expected QoS values of its component services.

UTILITY FUNCTION

In order to evaluate the multi-dimensional quality of a given web service composition a utility function is used. In this paper we use a Multiple Attribute Decision Making approach for the utility function: i.e. the Simple Additive Weighting (SAW) technique. The utility computation involves scaling the values of QoS attributes to allow a uniform measurement of the multi-dimensional service qualities independent of their units and ranges. The scaling process is then followed by a weighting process for representing user priorities and preferences. In the scaling process each QoS attribute value is transformed into a value between 0 and 1, by comparing it with the minimum and maximum possible aggregated value. These values can be easily estimated by aggregating the local minimum (or maximum) possible value of each service class in CS. For example, the maximum execution price of any concrete composite service can be computed by summing up the execution price of the most expensive service in each service class.

A SCALABLE QoS COMPUTATION

In this proposed work we present a scalable solution to the QoS-bases web service composition problem. We decompose the global optimization problem into sub-problems that can be solved independently. For this purpose, we first map the global QoS computation on the composite service level $U'(CS)$ into local computations that can be performed on each service class independently. Second, we propose a simple algorithm for decomposing each global QoS constraint $c'_k \in C'$ into n local constraints that can be verified locally on the component services. Finally, we present a distributed service selection algorithm that leverages local search for achieving global QoS requirements.

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

IARJSET

International Advanced Research Journal in Science, Engineering and Technology

ICRAESIT - 2015

B V Raju Institute of Technology, Narsapur

Vol. 2, Special Issue 2, December 2015

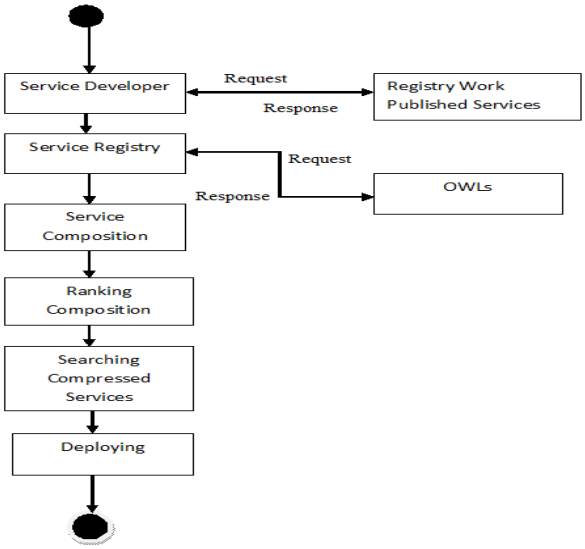

Figure 1: Scalable web service Composition

DECOMPOSITION OF GLOBAL QoS COMPUTATION

The use of (3) on the composite service level requires enumerating all possible combinations of the service candidates to find the optimal selection. This approach can be very inefficient for large scale problems or applications with run-time requirements. Therefore, we derive a modified utility function $U'_{local}(s)$ from $U'(CS)$ that can be applied on the component service level, without the need for evaluating all possible combinations.

$$U'(CS) = \sum_{j=1}^{n} \sum_{k=1}^{r} \frac{Q\max(i,j) - qk(s_j)}{Q\max'(k) - Q\min'(k)}.w_k$$

DECOMPOSITION OF GLOBAL CONSTRAINTS

To ensure that the outcome of the local QoS computation satisfies global QoS constraints, we need to decompose each global constraint $c'_k \in C', 1 \le k \le m$ into n local constraints. We use local statistics about the quality values to estimate a reasonable decomposition of each global constraint c'_k as follows:

Initially set the local constraint value c_{jk} of each service class to the local maximum value of that class,

$$\forall c'_k \in C' : d_k = \sum_{j=1}^{n} c_{jk} - c'_k$$

DISTRIBUTED OPTIMIZATION OF THE QoS COMPUTATION

We assume an architecture consisting of a service composer and a number of service brokers - either distributed or on a single machine. Each service broker is responsible for managing QoS information of a set of web service classes. A list of available web services is maintained by the service broker along with registered measurements of their non-functional properties, i.e. QoS attributes, like response time, throughput, price etc. The service composer instantiates a composite service CS in interaction with the service brokers.

$$\delta_k = \sum_{j=1}^{n} (c_{jk} - q_{jk}), 1 \le k \le m$$

III. EXPERIMENTAL EVALUATION

The model is considered by evaluating on a HP ProLiant DL380 G3 machine operating on Linux (CentOS release 5) as well as Java 1.6, with 2 Intel Xeon 2.80GHz processors also 6 GB RAM. In this evaluation the efficiency and the

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

IARJSET

International Advanced Research Journal in Science, Engineering and Technology

ICRAESIT - 2015

B V Raju Institute of Technology, Narsapur

Vol. 2, Special Issue 2, December 2015

quality of the outcomes of the model are reviewed with that of the linear programming techniques (LP) [4,5] also the heuristic algorithm WS_HEU [7]. In scenario of linear programming techniques, open source Linear Programming method Ipsolve version 5.5 [11] is utilized.

In scenario of WS_HEU an own execution is used. The execution is completed practically regarding each possible optimizations minimizing to the low the time of calculation. The assessments are performed with several situations of the QoS composition issue with modifications of the amount of service classes (n) also the amount of service applicants per class l. These constraints have unique pairing and the specific combination is displayed as one illustration of the composition concern. The QoS information the QWS real dataset in Table 1 is applied. The dataset consists of 9 QoS features specifications relevant to 364 real web providers. For an evaluation of the scalability issue a test according to bigger group of services is needed. To conquer this issue, the QWS dataset is tested on many times, multiplying the values of web providers' quality every time with a consistently dispensed random value between 0.1 as well as 2.0. With this strategy it is prospective of getting a data set of concerning 100.000 services.

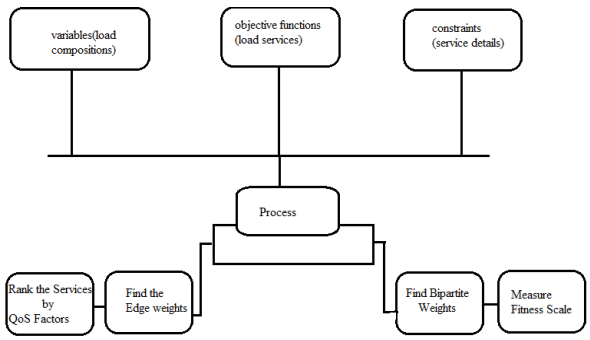

Figure 2 : Experimental Evaluation for Web Services

PERFORMANCE EVALUATION

The efficiency assessment of the preceding three techniques is completed by evaluating the time appropriate for choosing the remedy or the best pairing of tangible services of specific methods. In Fig 3, Performance analysis of automated service trace and composition strategies under composition accuracy metric, WS_HEU also the model we recommended DISTHEU is revealed. In our studies the efficiency of the three strategies is considered in regards to the size of the issue, amount of service classes (n) as well as the amount of service prospects per class l.

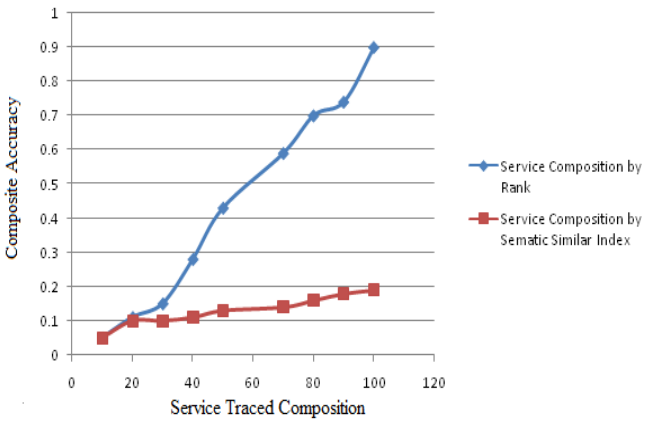

Figure 3 : Performance Analysis of Automated Service Trace and Composition Strategies Under Composition Accuracy Metric

OPTIMALITY EVALUATION

The Optimality is estimated with the optimality ration $R = Ua_{up}$. A review of the optimal outcomes of the strategy evaluated with that of LP technique is completed. The strategy evaluated in this document produces the utility of the best composition U_{approx} as per (3) also the LP technique produces the utility of composition U_{opt} . The outcomes revealed in fig 3 illustrate that DIST_HEU reaches best outcomes with an average of 98% optimality ratio. Furthermore the quality of the outcomes of the evaluated strategy DIST_HEU decreases by average, simply 1% below the outcomes of WS_HEU. The cost included is extremely high however for WS_HEU for this small calculation time advancement as observed.

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

IARJSET

International Advanced Research Journal in Science, Engineering and Technology

ICRAESIT - 2015

B V Raju Institute of Technology, Narsapur

Vol. 2, Special Issue 2, December 2015

Table 1: Optimality Evaluation using The Linear Programming Technique

Services	Fault Proness Scale	Variance of Fault Prone Scale	Standard Deviation of Fault Prone Scale	Lower Bond of Fault prone Scale	Upper Bond of Fault prone Scale
10	0.055	0.043	0.204	0.018	0.061
20	0.028	0.044	0.205	0.027	0.071
30	0.032	0.046	0.209	0.016	0.062
40	0.051	0.052	0.213	0.028	0.080
50	0.043	0.035	0.201	0.019	0.073
60	0.059	0.042	0.208	0.032	0.086
70	0.064	0.041	0.202	0.023	0.105
80	0.072	0.054	0.211	0.030	0.108
90	0.086	0.049	0.216	0.029	0.092
100	0.092	0.051	0.221	0.031	0.110

IV. CONCLUSION AND FUTURE WORK

The efficient web service discovery needs, the user must be able to discover all appropriate web services within the UDDI irrespective of the predefined categories, and all appropriate web services must be successfully discovered even if the user is not aware of all the relevant terms that include all appropriate web services. In this paper we have considered the semantic based ontology approach involves service categorization and selection of services with semantic service description and the composition of web service using OWL-S. In this regard, this work tends to actuate the requirement to integrate automated service composition. . We have tested the proposed approach by using a sample web service application. As future work, we extend to explore additional mapping tools to express service request to search for relevant concepts.

REFERENCES

[1] OASIS: Web services business process execution language (April 2007) http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf.
[2] Pisinger, D.: Algorithms for Knapsack Problems. PhD thesis, University of Copenhagen,Dept. of Computer Science (1995)
[3] Nemhauser, G.L., Wolsey, L.A.: Integer and Combinatorial Optimization. Wiley-Interscience, New York, NY, USA (1988)
[4] Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.Z.: Quality driven webservices composition. In: WWW. (2003) 411–421
[5] Ardagna, D., Pernici, B.: Adaptive service composition in flexible processes. IEEE Trans.Software Eng. 33(6) (2007) 369–384
[6] Liu, Y., Ngu, A.H.H., Zeng, L.: Qos computation and policing in dynamic web serviceselection. In: WWW. (2004) 66–73
[7] Yu, T., Zhang, Y., Lin, K.J.: Efficient algorithms for web services selection with end-to-endqos constraints. ACM Trans. Web 1(1) (2007) 6
[8] Maros, I.: Computational Techniques of the Simplex Method. Springer (2003)9. Li, F., Yang, F., Shuang, K., Su, S.: Q-peer: A decentralized qos registry architecture for webservices. In: ICSOC. (2007) 145–156
[9] Yoon, K..P., Hwang, C.L.: Multiple Attribute Decision Making: An Introduction (Quantitative Applications in the Social Sciences. Sage Publications (1995)
[10] Michel Berkelaar, Kjell Eikland, P.N.: Open source (mixed-integer) linear programming system. Source forge http://lpsolve.sourceforge.net/.
[11] Al-Masri, E., Mahmoud, Q.H.: Investigating web services on the world wide web. In: WWW.(2008)