Optimal QoS Aware Multi-Path Web Service Composition Using Heuristics Algorithm And Data Mining

Osama K. T. Qtaish[1], Zulikha Jamaludin[2] and Massudi Mahmuddin[3]
School of Computing, College of Arts and Sciences,
Universiti Utara Malaysia, 06010, UUM Sintok, Malaysia
[1]oqtaish@yahoo.com, [2]zulie@uum.edu.my, [3]ady@uum.edu.my

ABSTRACT

In QoS aware service composition, specifically in multiple execution paths compositions, how can we produce optimal composite services that satisfy QoS requirements supplied by clients? Can we generate a solution that simultaneously optimizes all the execution paths, while meeting global QoS constraints imposed by the clients? This paper proposes an optimization mechanism that optimizes the execution paths of the followed path during the execution. The problem is modeled as multi-dimensional multi-choice knapsack problem (MMKP) and a heuristic algorithm is also applied. Data mining is used to predict path at runtime in an attempt to generate the best possible solution.

Keywords: QoS, web service composition, data mining, heuristic algorithm.

I BACKGROUND

Web services are the leading technology and an emerging paradigm in service oriented computing for implementing Service Oriented Architecture (SOA). This new paradigm is ensuring rapid development, low in cost, platform-independent, self-described interface, and loosely couple software system. Web services are software systems designed to perform functionality. The web services are published by service providers (normally any organizations that provide service descriptions and ensure service implementations), and invoked by clients (organizations) without the client need to install as what in the server (Hilari, 2009).

One of the main benefits gained from implementing web services and SOA is the ability to compose new functionality out of existing outsourced web services into web service composition (Rosenberg et al., 2009). SOA along with service composition technology have changed the way software engineers design and develop business processes. Rather than developing entirely new processes, SOA processes are developed by composing network available web services (Dustdar & Papazoglou, 2008).

Business process is comprise of a set of related tasks or activities that been designed to fulfill a specific goal. Each task (also known as abstract web service) can be accomplished by a single outsourced web service hosted by external partners. For example a software engineer of online bank loan application, in the design time defines the business process by identifying and arranging the abstract services. Based on the semantic descriptions of the abstract services, many functionality equivalents web services (similar services functionality called candidates) can be discovered for each abstract web service. Then, the service selection can be performed dynamically at runtime by selecting the best outsourced services that can accomplish the abstract services' functionality (Ardagna & Pernici, 2006). One of the most substantial selection factors that can be served as selection criteria between those equivalent services is the QoS criteria including web service's non-functional characteristics such as cost, response time, availability, and reliability. Fig. 1 illustrates three different possible execution path(s) *EP* of the services *S ={S1, S2,…,S9}* to meet the global goal of abstract web service *AS*.

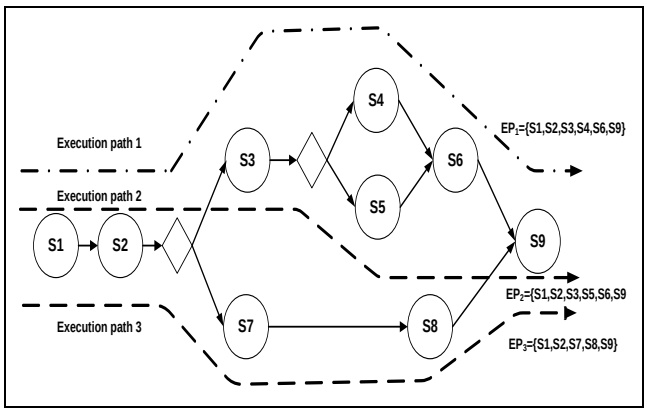

Figure 1. Multiple Execution Paths Composition.

$$EP_1 = \{S_1, S_2, S_3, S_4, S_6, S_9\}$$
$$EP_2 = \{S_1, S_2, S_3, S_5, S_6, S_9\}$$
$$EP_3 = \{S_1, S_2, S_7, S_8, S_9\}$$
$$\text{and } EP_1, EP_2, EP_3 \subset AS$$

However, it is difficult to generate a solution that simultaneously optimizes all execution paths involved in the composition at the same time, and satisfy global QoS constraints imposed by clients. Consequently, current approaches are not able to always guarantee generating an optimal/near-to-optimal solution such that the selection for each of the execution paths in the composition is optimal. For service optimization, finding an exact optimal solution required a strategy based on evaluating all the possible combinations to find the optimal one. Such a straightforward strategy results in a combinatorial problem, and the computational complexity to find a solution for this problem is non-deterministic polynomial time (Garey & Johnson, 1979). It is impractical and time consuming to evaluate all these combinations to find the optimal one. Therefore, solutions based on heuristic methods, although they deliver near-to-optimal solutions, are desired (Jaeger, Muhl, & Golze, 2007).

The aim of this work is to tackle this major issue occurs when optimizing business processes including multiple execution paths. For this purpose, an innovative QoS-aware service composition approach is proposed that efficiently suits for optimizing multi-path compositions. In this approach, an optimization mechanism is proposed based on the combination between data mining method and heuristic algorithms. This mechanism computes the optimization by considering only the path that will potentially be followed during the execution of a business process.

II RELATED WORKS

There are many different techniques are proposed to handle optimizing compositions containing multiple execution paths. Yu, Zhang and Lin (2007), Canfora et al. (2005), Wang, Chi, and Deng (2009) and Lécué (2009) are working on computation of certain path that more likely to be executed than another according to probability or stochastic information of paths execution. Estimation of the paths' probability of executions is estimated either by inspecting the system logs or specified by the composition engineers. However, such approaches might produce incorrect result. As a consequence, the generated solutions may not have the best QoS performance, worse than that; it may violate constraints imposed by the clients.

Ukor and Carpenter (2008, 2009) optimize each path separately by decomposing the composition into execution paths. After the optimization process, the execution paths are aggregated into an overall composition that consists of all paths. If there is a common abstract service that belongs to more than one path, the system identifies the hot path for the considered web service. They define the hot path as the path that has been most frequently used to execute the considered service. However, in the case that the actual execution of the composition is not following the hot path, the executed path may not have the best performance. Worst case, the executed path may not meet global QoS constraints.

Uker and Carpenter (2008) survey the QoS-aware service composition literature approaches in order to analyse the ability of the optimization algorithms to simultaneously generate optimal plans for all executions paths involved in compositions. They conclude that, within the context of multi-path compositions, it is difficult to generate a solution that simultaneously optimizes all the execution paths in the composition at the same time.

In a subsequent paper (Ukor & Carpenter, 2009), the authors address this problem by presenting an approach that enables users to bias the optimizations using a set of meta-metrics including execution probability of an activity, previous execution history of each activity, and probability of occurrence. The approach aims to find an approximation solution for each path involved in the composition. A trade-off between the paths is made, which chooses a path to favor by using a set of meta-metrics. For each path, the meta-metrics are computed as the weighted average of the aggregate values of meta-metrics. Then, the selection problem is solved using integer programming solution. Recent work that similar to this approach by Neelavathi and Vivekanandan (2011), proposed to use meta-metrics to resolve conflicts caused by optimizing multiple execution paths. Their meta-metrics represent a priority of execution activities in execution paths. However, these approaches suffer from several drawbacks. First, they compute the optimization by considering all the execution paths involved in the composition which may result in suboptimal solution for some paths. Second, the service selection optimization is biased using a set of meta-metrics. These meta-metrics are based on assumptions either assigned by the composition developers or estimated from the log trace records. These assumptions could be wrong. Consequently, the solutions obtained from these approaches may prove to be suboptimal for some execution paths.

In contrast to the above mentioned approaches, the solutions generated from our approach are expected to deliver the best possible QoS, at the same time, satisfy global constraints. This is because of our optimization mechanism, which designed to optimize only the path that will be followed during the execution.

III MULTI-CHOICE KNAPSACK PROBLEM ON WEB SERVICES

Multi path web composition is look similar to multi dimension multi-choice knapsack problem (MMKP). Lets there are N object groups whre each has $I_i (1 \le i \le N)$ objects. Each of these objects has profits p_{ij} and requires resource $r_{ij} = (r1_{ij}, ..., rm_{ij})$. The amount of resources available in the knapsack is $R = (R1, ..., Rm)$. MMKP is to select exactly one object from each object group to be placed in the knapsack so that the total profit is maximized while the total resources used are less than the resources available.

On the other hand, QoS selection aims to select exactly one candidate from each service class such that the entire QoS value of the composition is optimized while QoS requirements defined by clients are satisfied. Building on the similarity between these two problems, the selection problem can be mapped to MMKP as follows:

- The knapsack is represented by the composition.
- Each service class represents an object group.
- Each candidate in a service class represents one object in a group.
- Each utility function u_{ij} represents a profit p_{ij} and can be calculated using Eq.1.
- The QoS characteristics q_{ij} of a candidate s_j represent the required recourse r_{ij} of an object.
- The QoS global constraints *GS* are considered as the resources available in the knapsack *R*.

The algorithm is to select exactly one candidate from each service class such that the entire QoS value of the composition is optimized while QoS requirements defined by clients are satisfied. However, the MMKP requires that the total resources are less than the recourse available. But in the selection problem, the total QoS characteristics are required to be either less (for negative characteristics) or greater (for positive) than the global QoS constraints. Therefore, positive characteristics are needed to be transformed into negative. To do so, the values of positive characteristics are multiplied by -1.

IV MULTI-PATH QOS-AWARE SERVICE COMPISITION

As illustrated in Fig. 2, our proposed QoS-aware composition process involve a few activities before the desired optimal composition plan is attained, namely i) abstract composition ii) discover paths that contains a list of outsourced candidate web services discovered for each abstract service, iii) path prediction and iv) QoS optimal computation based on QoS characteristic values.

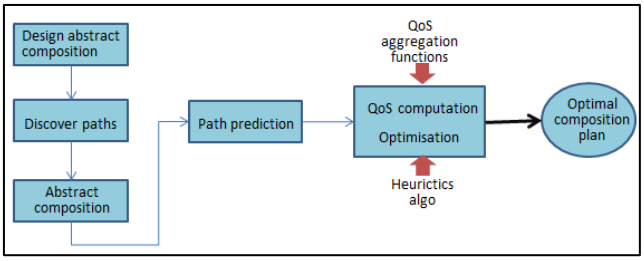

Figure 2. Proposed Approach.

A. Abstract Composition

In this work, the QoS-aware compositions process is carried out dynamically on instance by instance basis. It is started by predicting, at runtime, and just before the actual execution of compositions, the path that potentially will be followed during the execution of a composition. Note that this step will be carried out offline, so it does not affect the performance of the proposed approach.

Additional data are required to be recorded in logs to perform execution path prediction. These data are runtime generated information indicating the input (output) values parameters passed (received) to (from) web services and their types. These values are generated at runtime during the execution of composition instances. Each 'parameter/value' entry has a data type, a name, and a value to store such information, the current composition logs need to be extended. Furthermore, an extra field needs to be added to the log in order to store execution path information; which describing the path that has been taken during the execution of a composite service (i.e., the web services that have been executed).

B. Selection Criteria

In our approach, we considered the following QoS characteristics as selection criteria: cost, response time, reliability, availability, security (encryption level), throughput, reputation and composability. These QoS characteristics are identified based on our previous work (Qtaish & Jamaludin, 2011), where we investigating and analyzing 25 QoS characteristics that were used in the area of web service and SOA.

Cost represents the amount of money that a service requester has to pay for a service provider for using its service. Response time is a typical measure of performance. It represents the total time required to complete a service request, which can be defined by the sum of the time a service need to process a request on the provider side (processing time), and the time

needed to send a request and receive a response over a network (Lee at al., 2012). For reliability, it represents degree that a service is able to correctly respond to a request in a specified time interval. The number of a service's failures in minutes, days or months describes its reliability (Mani & Nagarajan, 2002). The availability of a web service represents the probability that the service is ready for access when required to immediate use (Zeng et al., 2004; Mani & Nagarajan, 2002). Security characteristic can include numerous aspects. It means providing confidentiality, authentication, authorization, encrypting data and non-repudiation. These security aspects can be provided in different level of policy by service providers (Lee et al., 2012; Mani & Nagarajan, 2002). The only aspect that can be described with numerical value is the encryption level, which is a measurement that describes the length for a key that is used for encryption (Guoping, Huijuan, & Zhibin, 2009). For throughput, it is a measure of service's productivity. It can be defined as the number of requests that the service provider can process in a given time period (Lee et al., 2012; Mani & Nagarajan, 2002). Reputation represents a ranking that is provided by the users of a service based on their experience of using it. Reputation measures the trustworthiness of a service (Zeng et al., 2004). Finally, composability represents the probability that the service is executed as a member of the composition service (Guoping, Huijuan, & Zhibin, 2009).

C. QoS Aggregation for Web Service Composition

The QoS value of a composite service is computed from its constituent web services. The QoS value of a composite service SN is defined by the vector Q. This tuple contains the aggregated QoS values of a composite service (i.e., a solution) represented by the index *k*, $Q=(Q_1(SN),\ldots,Q_n(SN))$, where $Q_k(SN)$ is the estimated k[th] QoS characteristic of the composite service SN. The aggregation functions are presented in the Table 1. Some aggregation functions are similar to those proposed by Zeng et al. (2004) and Jaeger et al. (2004). In the table, the variable *a* represents the number of services involved for the computation. The variable x_{ij} represents a selection variable. Note that the web services that only belong to the predicted path are considered for computation i.e. $i \in EP_{pred}$.

Table 1. QoS Aggregation Functions.

QoS Characteristic	Aggregation Functions
Cost	$Q_k(SN)=\sum_{i\in EP_{pred}}^{k} q_{ij} * x_{ij}$
Response time	$Q_k(SN)=\sum_{i\in EP_{pred}}^{k} q_{ij} * x_{ij}$
Reliability	$Q_k(SN)=\prod_{i\in EP_{pred}}^{k} q_{ij} * x_{ij}$
Availability	$Q_k(SN)=\prod_{i\in EP_{pred}}^{k} q_{ij} * x_{ij}$
Encryption level	$Q_k(SN)=\min(\overset{k}{q_{ij}} * x_{ij}), \forall i \in EP_{pred}$
Throughput	$Q_k(SN)=\min(\overset{k}{q_{ij}} * x_{ij}), \forall i \in EP_{pred}$
Reputation	$Q_k(SN)=\frac{\sum_{i\in EP_{pred}}^{k} q_{ij} * x_{ij}}{a}$
Composability	$Q_k(SN)=\frac{\sum_{i\in EP_{pred}}^{k} q_{ij} * x_{ij}}{a}$

As presented in Table 1, availability and reliability are multiplicative QoS characteristics (aggregated as a product) and result in nonlinear functions. However, linear functions (aggregated as a summation) are easier to be processed. Therefore, similar to Zeng et al. (2004), nonlinear functions are transformed into linear by applying a logarithm operation.

D. Utility Function

If more than one QoS characteristics are subject to optimization, then an aggregated goal function is needed. For this purpose, each candidate service $s_j \in S_i$ is associated with utility function u_{ij}. This function depends on the QoS characteristics' types i.e., positive or negative. For positive QoS characteristics, meaning that a higher value denotes a better QoS, like availability and reliability, the function needs to maximize the values, whereas, the values of the negative characteristics, like cost and response time, needs to minimize. In addition, QoS characteristics have different units of measurements. For example, reliability is a probability ratio and varies between 0 and 1 while response time is expressed in milliseconds by a positive number. To perform fair computation, and to allow uniform measurement of different QoS characteristics independent from their units and ranges, all the QoS characteristics are normalize by their average and slandered deviation. Furthermore, all the QoS characteristics are weighted by their importance. The utility function is defined as follows (Yu, Zhang, & Lin, 2007):

$$u_{ij}=\left(\sum_{k=1}^{2}\left(\frac{\overset{k}{\mu_i}-\overset{k}{q_{ij}}}{\overset{k}{\sigma_i}}\right)*W_k+\sum_{k=3}^{8}\left(\frac{\overset{k}{q_{ij}}-\overset{k}{\mu_i}}{\overset{k}{\sigma_i}}\right)*W_k\right) \quad (1)$$

where W_k is the weight assign for each QoS characteristics which defined by clients such that

$W_k \in [0,1]$ and $\sum_{k=1}^{n} W_k = 1$. The index $k = 1 \ldots 8$ represents the different QoS characteristics. The first part of the equation i.e., the summation where $k = 1,2$ represents the normalization of the negative QoS characteristics, while the second represents the summation for the positive. μ_i and σ_i are the average and standard deviation for the values of QoS characteristics of all candidates in service class S_i.

The selection variables x_{ij} is used to determine whether a candidate service is selected for optimal composition or not. The value of x_{ij} is either 0 or 1. 1 if the candidate service s_{ij} is selected for the abstract service as_i, while 0 if not. There is exactly one candidate service selected for each abstract service as_i i.e., $\forall i, 1 \le i \le a, x_{i1} + x_{i2} + \ldots + x_{ib_i} = 1$.

Based on the above, the selection problem can be modeled as follows:

$$\max\left(\sum_{i=1}^{1}\sum_{j=1}^{b_i} u_{ij} * x_{ij}\right)$$

Subject to the global QoS constraints

$Q_k(SN) \le \overset{k}{GS}$, for negative QoS characteristics (price and response time)

$Q_k(SN) \ge \overset{k}{GS}$, for positive QoS characteristics (rest of QoS characteristics)

while keeping:

$\sum_{j=1}^{b_i} x_{ij} = 1$, $\forall i \in \{1,\ldots, a\}$, and

$$x_{ij} = \begin{cases} 1, & if \quad s_{ij} \quad is \quad selected \\ 0, & if \quad s_{ij} \quad is \quad not \quad selected \end{cases}$$

V PATH PREDICTION

Apart from the QoS-aware service composition, Cardoso (2008) also emphasizes the importance of QoS management for workflows and organizations. He focusses on predicting the QoS of workflows before they executed or during the execution. To this aim, the author proposes a data mining technique that allows predicting with high level of accuracy the QoS of workflows. The method is adapted here for the purpose of predicting, at runtime, the sequence of web services that will be executed during a composition's execution.

A. Learning Phase Method

The recorded composition log data is cleaning using data mining instances by extracting the input/output values of web services as well as the path information from the log. The extracted data are converted to a suitable format of machine learning algorithms. Each instance is characterized by the values of six input/output parameters of web services and associated with a class, indicating the path that has been followed during the execution when the parameters of web services have been assigned to a specific value set. In summary, the instance structure is as follows:

> income, loan type, loan amount, loan years, name, SSN , [EP]

The parameter income, loan amount, loan years and SSN are numeric, whereas the attributes loan type and name are nominal. Once enough instances are created, it will constitute inputs to a machine learning algorithm to establish a relationship between the input/output parameters and the paths taken at runtime. A set of classified instances is taken by a learning schema to learn a way of classifying unseen instances. Since the class *EP* of each instances is provided, we uses supervised learning (Sumathi & Esakkirajan, 2007). In our approach, the algorithm will be trained offline so the performance of the approach does not be affected by the computation time needed for training the algorithm.

Different machine learning methods can be employed to carry out path prediction. Cardoso (2008) conducts a set of experiments using Naïve Base, J48, and Sequential Minimal Optimization methods with and without the Multiboost method. J48 algorithm is Weka's (Hall, 2009) implementation of the C4.5 (Quinlan, 1993) decision three learners. Naïve Bayes classifier technique is based on the so-called Bayesian theorem. SMO [28] is a fast method to train Support Vector Machines (Cortes & Vapnik, 1995). MB L. (Todorovski, & Džeroski, 2000) is an improved meta learning algorithm of AdaBoost (Freund & Schapire, 1999). Based on Cardoso's experiments, the results indicate that the Multiboost Naïve Base (MB NB) approach is the data mining algorithm that yields the best path prediction accuracy results (Sumathi & Esakkirajan, 2007). Therefore, this work employs the MB NB as well.

B. Path Prediction Method

The adapted data mining method is composed of three steps, the first two are similar to Cardoso (2008), while the third is added to the purpose of runtime predication, and carried out at runtime. Once the data is formatted and analyzed by a learning method, it is ready for classifying unknown classes, i.e., predict the path that will be followed during the execution. At

runtime, a client (such as a service requester) is required to provide data including personal data and data describing the condition of the service being requested. Then, the data needed for prediction are collected and formatted, and then input to a classifier to classify into target classes, i.e., composition paths.

The output of this step is the predicted probability of a given execution path EP_i will potentially be followed during the execution of the bank loan composite service. This important information is then utilized by the optimization algorithms in order to only optimize the predicted execution path.

VI CONCLUSION

The optimization algorithm proposed takes into account the probability of the predicted execution path. This way the composition is expected to always generate solutions with the best QoS metrics and also fulfill the global constraints imposed by clients.

REFERENCES

Ardagna, D. & Pernici, B. (2006). Global and Local QoS Guarantee in Web Service Selection, in C. Bussler and A. Haller (Eds.), *Business Process Management Workshops*, 3812, Heidelberg: Springer Berlin, 32-46.

Canfora, G., Penta, M. D., Esposito, R., & Villani, M. L. (2005). An approach for QoS-aware service composition based on genetic algorithms. *Paper presented in 2005 conference on Genetic and evolutionary computation, Washington*, ACM. 1069-1075.

Cardoso, J. (2008). Applying Data Mining Algorithms to Calculate the Quality of Service of Workflow Processes, in P. Chountas, I. Petrounias & J. Kacprzyk (Eds.), *Intelligent Techniques and Tools for Novel System Architectures*, 109. Heidelberg: Springer Berlin. 3-18.

Cortes, C. & Vapnik, V. (1995). Support-vector networks, *Machine Learning*, 20(3), , 273-297.

Dustdar, S. & Papazoglou, M. P. (2008). Services and Service Composition - An Introduction, it - Information Technology, 50, 086 - 092.

Freund, Y. & Schapire, R. E. (1999). A Short Introduction to Boosting. *Journal of Japanese Society for Artificial Intelligence*. 14(5), 771-780.

Garey, M. R. & Johnson, D. S. (1979). *Computers and Intractability: a Guide to the Theory of NP-Completenes.* New York: W. H. Freeman & Co.

Guoping, Z. Huijuan, Z. & Zhibin, W. (2009). A QoS-Based Web Services Selection Method for Dynamic Web Service Composition, Proc. 2009 First International Workshop on Education Technology and Computer Science, Hubei, IEEE Computer Society, , 832-835.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations*. 11(1).

Hilari, M. O. (2009). *Quality of Service (QoS) in SOA Systems. A Systematic Review*, Master thesis, Universitat Politècnica de Catalunya.

Jaeger, M., Muhl, G. & Golze, S. (2007) QoS-Aware Composition of Web Services: An Evaluation of Selection Algorithms, in R. Meersman and Z. Tari (Eds.), *On the Move to Meaningful Internet Systems 2005, 3760*. Heidelberg: Springer Berlin. 646-661.

Jaeger, M. C., Rojec-Goldmann, G., & Muhl, G. (2004). QoS aggregation for Web service composition using workflow patterns. *Paper presented in Enterprise Distributed Object Computing Conference, Eighth IEEE International*. Washington, IEEE Computer Society, 149-159.

Lécué, F. (2009). Optimizing QoS-Aware Semantic Web Service Composition, in Bernstein, A., Karger, D., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., & Thirunarayan K. (Eds.). *The Semantic Web - ISWC 2009, 5823*. Heidelberg: Springer Berlin. 375-391.

Lee, K., Jeon, J., Lee, W., Jeong, S.-H., & Park, S.-W. (2012). *QoS for Web Services: Requirements and Possible Approaches, W3C Web Services Architecture Working Group*. Tech. Rep., November 2003. [Online]. Available: http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/ last accessed Feb.

Mani, & Nagarajan, A. (2002). Understanding quality of service for Web services, IBM developerWorks, Tech. Rep., January 2002. [Online]. Available: http://www.ibm.com/developerworks/webservices/library/ws-quality/index.html last accessed Feb, 2012.

Neelavathi, S., & Vivekanandan, K. (2011). An Innovative Quality of Service (QOS) based Service Selection for Service Orchrestration in SOA. *International Journal of Scientific and Engineering Research*. 2(4).

Quinlan, J. R. (1993). *C4.5: Programs for machine learning.* (San Francisco: Morgan Kaufmann Publishers Inc.

Qtaish, O. K., & Jamaludin, Z. B. (2011). QoS criteria for distinguishing the competing web services. *Paper presented in 2011 First IRAST International Conference on Data Engineering and Internet Technology (DEIT).* Bali, IEEE Computer Society.

Rosenberg, F., Leitner, P., Michlmayr, A., Celikovic, P. & Dustdar, S. (2009,). Towards Composition as a Service - A Quality of Service Driven Approach. Proc. 2009 IEEE International Conference on Data Engineering, IEEE Computer Society, 1733-1740.

Sumathi, S., & Esakkirajan, S. (2007). *Fundamentals of relational database management systems.* Heidelberg: Springer Berlin.

Todorovski, L. & Džeroski, S. (2000). Combining Multiple Models with Meta Decision Trees, in D. Zighed, J. Komorowski & J. Zytkow (Eds.). *Principles of Data Mining and Knowledge Discovery, 1910*. Heidelberg: Springer Berlin. 69-84.

Ukor, R. & Carpenter, A. (2008). On Modelled Flexibility and Service Selection Optimisation. *Paper presented at 9th Workshop on Business Process Modeling, Development and Support*. Montpellier, 335.

Ukor, R., & Carpenter, A. (2009). Flexible Service Selection Optimization Using Meta-Metrics. *Paper presented in 2009 Congress on Services – I.* IEEE Computer Society. 593-598.

Wang, R., Chi, C.-H., & Deng, J. (2009). A Fast Heuristic Algorithm for the Composite Web Service Selection, in Li, Q., Feng, L., Pei, J., Wang, S., Zhou, X., & Zhu, Q.-M. (Eds.), *Advances in Data and Web Management, 5446.* Heidelberg: Springer Berlin. 506-518.

Yu, T., Zhang, Y., & Lin, K.-J. (2007). Efficient algorithms for Web services selection with end-to-end QoS constraints. *ACM Transactions on the Web (TWEB)*. 1(1), 6.

Zeng, L., Benatallah, B., Ngu, A. H. H., Duma, M., Kalagnanam, J. & Chang, H. (2004). QoS-aware middleware for Web services composition. *IEEE Transactions on Software Engineering*. 30(5), 311-327.