QoS-Aware Web Service Composition Using Kernel-Based Online Quantile Estimation

Dries Geebelen[*] [†]
dries.geebelen@esat.kuleuven.be

Kristof Geebelen[*] [‡]
kristof.geebelen@cs.kuleuven.be

Eddy Truyen[‡]
eddy.truyen@cs.kuleuven.be

Joos Vandewalle[†]
joos.vandewalle@esat.kuleuven.be

Johan A.K. Suykens[†]
johan.suykens@esat.kuleuven.be

Wouter Joosen[‡]
wouter.joosen@cs.kuleuven.be

[†]ESAT-SCD-SISTA, Dept. Electrical Engineering, K.U.Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
[‡]IBBT-DistriNet, Dept. Computer Science, K.U.Leuven, Celestijnenlaan 200A, B-3001 Leuven, Belgium

ABSTRACT

Often many available web services provide similar functionality, but have different Quality of Service (QoS). In this case, a choice needs to be made to determine which services are selected to participate in a given service composition. This paper proposes a technique for QoS-aware service composition with the purpose to pro-actively minimize the rate of violations of a service level agreement (SLA) between a customer and the service (composition) provider. In contrast to most existing work on QoS based service composition, we assume time-varying QoS attributes of participating services and use time series prediction to determine an optimal composition. Due to the nature of the predication problem, we propose the use of a kernel-based online quantile estimator. Kernels enable the modeling of non-linearities in QoS attributes with respect to time. Online learning lowers the computational complexity compared to batch learning and allows the model to be easily updated with each new training example. Quantile estimation permits to quantify the violation rate of SLAs. We demonstrate the feasibility of this technique in an existing framework for on-demand service composition. This framework enables a service provider to offer to a customer on-the-fly creation of a customized BPEL process based on an SLA requested by the customer. Our results show that the proposed algorithms drastically reduce the violation rate of the SLA while minimizing the rejection rate of customer SLA requests.

Categories and Subject Descriptors

H.3.5 [**Online Information Services**]: Web-based services; G.3 [**Probability and Statistics**]: Time series analysis

General Terms

Management, Performance, Measurement, Algorithms

[*]Both authors contributed equally to this work.

Keywords

QoS; SLA; Service Composition; Time Series Prediction; Kernels; Online Learning; Quantile Estimation; WS-BPEL

1. INTRODUCTION

As Papazoglau et al. [1] explain in their report on longer term research challenges in Software & Services, globally available software services are expected to create opportunities for new business models that will take advantage of flexible, combinable sofware-services. There is a need for technologies allowing to effectively shift from inherently static processes and relatively simple service compositions evolving relatively over time, to global service networks that are complex and highly dynamic and which are actually evolving during execution to meet previously unknown requirements. An important challenge is that future service-based applications will require the capability to change dynamically the levels of quality of service they demand and to become truly responsive.

In the context of global service networks, Software as a service (SaaS), sometimes referred to as "Software on Demand", is an emerging mechanism of releasing software applications to customers. SaaS fits into the Could Computing paradigm and is differentiated by the fact that it can be implemented rapidly and eliminates the infrastructure and ongoing costs that traditional applications require. Details are abstracted from the users, who no longer have need for expertise in, or control over, the technology infrastructure "in the cloud" that supports them. Most cloud computing infrastructures consist of services delivered through common centers and built on servers. Clouds often appear as single points of access for all consumers' computing needs. Commercial offerings are generally expected to meet quality of service (QoS) requirements of customers, and typically include service level agreements (SLAs).

From a technical perspective, an import issue in "software on demand" is the customization for the SaaS application to serve multiple tenants. In this paper, we address customization of business processes for run-time assurance of service quality and service level agreements. A business process is a collection of related, structured activities or tasks that produce a specific service by combining services provided by multiple business partners. For example, an integrated travel planning web service can be created by composing services for hotel booking, airline booking, payment, etc.

More specifically, we focus on the Web Services Business Process Execution Language (WS-BPEL)[1], a workflow language that has profiled itself as the de-facto industry standard for orchestrating web services.

Several works on QoS-aware service composition considers QoS attributes of atomic services, such as response time or throughput, as deterministic quantities [2, 3, 4]. They utilize a pre-specified model of the process environment where composition solutions are based on point estimates, such as average values. Other approaches [5, 6] recognize that Business environments can be highly dynamic. This implies that there is no guarantee for services selected at design time that their quality properties have not changed at run-time. As a result, QoS values are regarded as stochastic, and the service composition problem is treated as a decision problem under uncertainty. To the best of our knowledge, there is no prevalent work that views composition optimization from the perspective of predictable time-varying QoS attributes in which can be verified whether the chance on violating a SLA is lower than a predefined value.

In this paper, we present a new approach for QoS based Web Service composition on demand for WS-BPEL. A tenant chooses a composition template using abstract services according to his functional requirements. Concrete service are chosen according to the SLA and the predicted QoS values of the participating services. Due to the dynamic nature of services and service-based applications, our approach validates and verifies SLAs at run-time. A monitor continuously observes for changing QoS values. When a potential SLA violation is predicted, the running business process will dynamically be adapted by choosing alternative services to suit the SLA again. For this purpose, we elaborated an algorithm that predicts the chance that a service composition will comply with the service level agreement. The algorithm uses kernel-based quantile estimation. We show how this technique can improve run-time assurance of service quality compared to techniques that use average values for QoS.

The remainder of this paper is organized as follows: Section 2 clarifies the problem statement by a motivating scenario. Section 3 provides and overview of the QoS model and presents the service composition framework. Section 4 describes the underlying prediction mechanism that is used to find a concrete service that can comply with the users SLA.

[1]`http://docs.oasis-open.org/wsbpel/`

An experimental evaluation of the QoS prediction technique is documented in Section 5. We discuss related work in Section 6. Finally, section 7 concludes the paper.

2. MOTIVATING SCENARIO

This section presents a case study situated in the health care environment. The case study consists of a workflow, initiated by the government, that realizes a mammography screening program in order to reduce breast cancer mortality by early detection for women above a certain age. The workflow is illustrated in Figure 1.

The first task of the workflow consists of sending out invitations to all women that qualify for the program to let a radiologist take images needed for the screening. Once images are taken, the radiologist uploads them to the system (task 2). Next, the image needs to be analyzed by specialized screening centers. There are always two independent readings, represented by tasks 3 and 4. These readings can be performed in parallel. In a next step, the two results of the readings are compared. When the results are identical, there is little doubt that the two physicians made the same mistake. Therefore it can be safely assumed that results are correct and the workflow can proceed with task 5. However, when the results are different, a concluding reading is performed (task 4'). Once the results of the screening of a particular screening subject are formulated, a report is generated (task 5) and different parties are billed (task 6). Task 5 and 6 can also be done in parallel and the billing task is not a necessity to continue the workflow. Finally, a report is sent to the screening subject and her general practitioner in task 7.

Suppose the government (tenant), who finances this initiative, wants some quality guarantees and specifies a service level agreement with the company (service provider), that is responsible for executing the workflow. In this agreement, the company specifies that in 97% of the cases the duration between task 2 and task 7 will take no longer than 5 working days. For simplicity, we only discuss the case in which the screening results are identical and task 4' does not have to be executed. To meet the total response of 5 working days, we demand task 3 and task 4 to be executed in 2 days and task 7 in 3 days with a probability higher than 99%. The duration of task 5 can be neglected since it is automated and takes only a fraction of time compared to the other tasks.

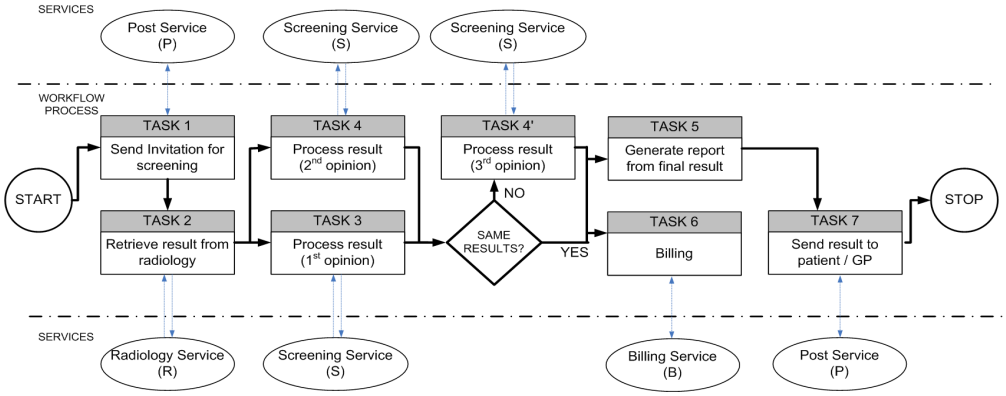


Figure 1: Example E-Health workflow process

If the chances on violation are uncorrelated for the three tasks, then the chance none of the three tasks violates the constraint equals the agreed violation rate of 3%. We will now show how a prediction algorithm in combination with quantile estimation could improve the service composition process for this simple scenario.

2.1 Time Varying vs. Time Independent QoS Prediction

Services	Monitored RT (days)				Real RT (days)		Average RT (days)
	t1	t2	t3	t4	t5	t6	
SS1	3	1	3	1	**3**	1	2
SS2	2	2	2	2	**2**	2	2
SS3	5	5	5	1	**1**	1	3
P1	3	3	3	3	3	**3**	3

Table 1: Monitored, real and average response times (RTs) for the screening and post services.

Suppose there are three potential candidate screening services for task 3 and 4 called screening service 1 (SS1), screening service 2 (SS2) and screening service 3 (SS3) for convenience. SS1 has an average response time (RT) of 2 days, but in reality RTs can vary as will be illustrated later with a real-life example. Possible causes are temporary over -or underload, infrastructure failures, etc. Suppose the response time for SS1 is volatile: RTs of 1 and 3 days occur often. SS2 has an average response time of 2 days as well, but is much more consistent. SS3 has a an average RT of 3 days, also with a high volatility. The post service (P1) is the only available service and has a stable RT of 3 days. The monitored and real RTs for these services on different time steps are summarized in Table 1.

Several works on QoS based web service composition do service selection based on point estimates such as time independent average values. In this simple scenario, this would imply that SS1, SS2 and P1 are selected to execute the workflow since they compose the optimal average workflow. SS1 and SS2 are executed in parallel at time t5 with an average RT of 2 days and P1 is executed on t6 with an average RT of 3 days. This composition takes in total 5 days on average. Now, if we take a look at the real values on t5 and t6, total duration is 6 days (max(2,3) and 3) and the SLA will be violated. A solution to this problem is to predict the RTs on t5 and t6 by using the monitored values of t1 to t4. Ideally, such an prediction algorithm would detect the patterns in SS1 and SS3 and estimates QoS attributes for each service close to the real values. The optimal composition would then be SS2 and SS3 in combination with P1 with a real total duration of 5 days. The government would be pleased that a valid composition can be identified at front of which the SLA is indeed respected in practice. The algorithms that we propose to achieve this goal are discussed in Section 4.

2.2 Quantile vs. Average SLA Compliance Estimation

We explain the benefit of quantile estimation by means of 2 candidate post services for task 7. Suppose the response time of these services have probability densities for RTs 1 to 5 days as presented in Table 2. We can use these values to make a quality estimate for a SLA. For example, which service would be the best candidate to have 99% certainty that its response time will not exceed 3 days? Using the average RTs of 3 days for P1 and 2 days for P2, it seems that P2 is more reliable than P1 and thus the best choice. However, using 99% quantile values of 3 days for P1 and 4 days for P2, we rightly conclude the opposite: P1 has a higher probability that it will not exceed the threshold of 3 days since it is less volatile.

Services	Monitored RT (Prob. Density)					99% Q-Value	Average Value
	1	2	3	4	5		
P1	0%	1%	98%	1%	0%	3	3
P2	15%	75%	6%	3%	1%	4	2

Table 2: Probability density, average values and 99%-quantile values of RTs for 2 post services.

3. COMPOSITION APPROACH OVERVIEW

3.1 QoS Criteria

In the domain of web services, QoS parameters can be used to determine non-functional properties of the service. QoS attributes can be divided into quantitative and qualitative attributes. Examples of the latter are security and privacy. They cannot be measured using metrics and will not be considered in the scope of this work. Popular quantitative attributes are response time, throughput, availability, reliability and cost:

- Response Time (RT): the time taken to send a request and receive a response (expressed in milliseconds). The response time is the sum of the processing time and the transmission time. For short-running processes they are usually of the same order. For long-running processes that can take hours, days or even weeks to complete, the transmission time is usually negligible.
- Throughput (TP): the maximum requests that can be handled at a given unit in time (expressed in requests/minute).
- Availability (A): the probability the web service is available (expressed in available time/total time)
- Cost (C): the cost that a service requester has to pay for invoking a specific operation of a service (expressed in cents/request)

The QoS of a service composition is calculated based on the QoS values of its constituents. In contrast to the measurement of QoS for elementary services, composite services consist of different activities such as sequences, if-conditions, loops and parallel invocations. We need to take into account these different composition patterns to calculate the QoS of a composite application. The optimization of the composition process, given the QoS attributes of its constituents has already been tackled in several works [2-6,8,9]. We will not address this issue in the scope of this paper, but only focus on achieving accurate QoS estimates of the atomic services. The potential impact of our proposed prediction algorithms on composition optimization is an interesting track for future work.

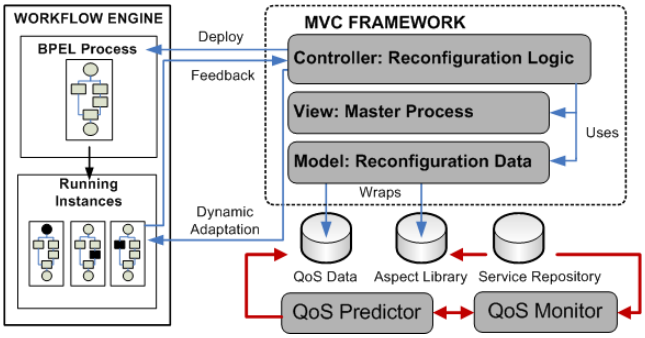


Figure 2: Framework overview

3.2 Composition Framework

In this section we present a framework that allows the adaptation of WS-BPEL processes in a dynamic and modular way. The idea is based on similar evolutions in the domain of web design. The intent of web design is to create a website that presents the content to the end users in the form of web pages. To comply with today's expectations of end-users, there is a growing tendency to use dynamic web pages. In contrast to static pages, where the content and layout is not changed with every request, dynamic pages adapt their content on the fly depending on the user's input. We map this concept of dynamic web design to web service composition. The proposed solution is based on the "Model-View-Controller" (MVC) concept.

3.2.1 Overview

An overview of our framework is illustrated in Figure 2. The framework generates a workflow process according to the adaptation logic specified in the controller component. This adaptation logic enforces SLAs for the composition by selecting appropriate services. The resulting process is a standard WS-BPEL process, deployable on existing WS-BPEL engines. Dynamic adaptation of running instances is done at run-time, for example, when during the execution new services need to be selected because the SLA threatens to be violated. During execution, the process instance feeds back to the framework to report its progress. The controller can then decide if adaptation is required. In the context of dynamic web pages, this approach is similar to generating web content based on the user's input. In this analogy, the XML representation of the workflow process can be interpreted as web content. We discuss the main building blocks of the framework: the master process, the controller, the data model and the QoS monitor and predictor.

Model - The model includes the aspect library and the QoS-related data. The library contains aspects for the different WS-BPEL activities that can be modularized as a specific task. An aspect definition represents for example a WS-BPEL fragment that contains the activities needed for the invocation of a specific web service. All these concrete implementations of a task are bundled in the library and can be reused across workflows. QoS-related data contains quality estimates of the available concrete services in the service repository.

View (Master Process) - With the aspect library in mind, a master process can be created. Instead of including all specific implementation details, it is designed as a template. The process is designed like a regular process and specifies the sequence of tasks that need to be executed. When the concrete implementation of a task depends on certain constraints, then only a general reference to the type of the task is included. Binding a concrete aspect to the task reference is done later by the controller according to the adaptation logic. In the context of QoS-aware service composition, an example abstract task is: invoke an airline booking service. A possible concrete implementation, called an aspect, is a WS-BPEL fragment for invoking Brussels Airlines' booking service.

Controller - The controller contains the logic to decide which aspects are substituted in the master process, i.e. it implements the adaptation policies that depend on the information available through the model. In the scope of this paper, a policy is defined as the SLA between service provider and tenant. Policies are enforced before deployment using QoS-related data and at run-time with dynamic adaptation mechanisms using the updated predicted QoS values and the current state of the workflow process. Concrete implementations for optimizing the composition process given the QoS values and SLA are not in the scope of this work. Many papers have already addressed the problem of service selection by local optimization and global planning. A frequently used technique to solve this problem is linear integer programming [2, 3].

QoS Monitor - Quantitative and time-varying QoS attributes for the available concrete services are collected by the monitor to compose time series that are used for predicting future values. Monitoring fully automated free services can be done by sending requests to the service and analyzing the results. Since too many service calls can impose overhead causing adverse effects on the service quality parameters like response time, it is important to choose acceptable time intervals. Long-running or costly processes require alternative methods for gathering QoS data.

QoS Predictor - This component uses the monitored data to make QoS predictions and quantile estimates that are used for validating SLAs of service compositions by the controller. QoS attributes are estimated for times when concrete services are planned to be executed. The controller then selects concrete implementations for the abstract tasks in the master process consistent with the SLA. The prediction algorithms we use for this purpose are explained in Section 4.

3.2.2 QoS-Based Composition

Our goal is to optimize service selection at deployment time, compliant with the service level agreement between service provider and tenant. Figure 3 shows the different steps of the composition process: (1) A tenant chooses a composition template based on abstract services and specifies his QoS needs in a service level agreement (2a) The controller selects the template from the compositions library, (2c) replaces the abstract services with concrete services available in the service repository. (2b) The selection is done using predicted QoS values from the QoS database.

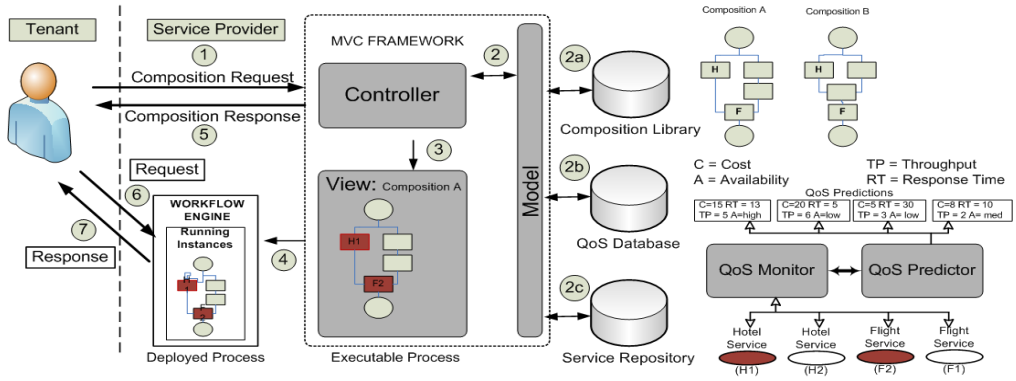


Figure 3: Composition framework

(3) If a suitable composition can be created, the resulting executable WS-BPEL process is generated by the controller and (4) deployed on the workflow engine. (5) The tenant will be informed if a composition with an acceptable chance on fulfilling the agreement has succeeded. (6 & 7) After receiving the service location, the tenant can now use his "composition on demand".

Also during the workflow execution, it is essential to optimize the composition according to the constraints implied by the service level agreement because the QoS values of the partner services can change in time. For this purpose, we use the same framework to perform run-time adaptation of WS-BPEL processes. In a first stage, the previous procedure is done to generate an optimal scenario at deployment time. If during the execution, the service agreement threatens to be violated, the MVC framework will try to replace a delaying service by another candidate. This is done on fixed points during execution in order to avoid inconsistent states. On these points, the process reports its state to the framework and asks to reevaluate the service agreement. Our implementation of this framework is an extension of the framework used in previous work [7]. A similar architecture was used to enforce workflow confidentiality in WS-BPEL. Due to space limitations, we refer to this work for a more detailed description of the underlying concepts of the framework and the prototype.

4. QOS PREDICTION

In this section we discuss methods to predict changes in QoS attributes. We propose two online quantile estimators, from which one is kernel-based, for the following reasons:

- We chose an online learning algorithm because the training phase of online learning methods generally takes less time and space compared to batch learning methods. The learned model can also easily be updated with each new training example.
- Quantile estimation determines whether the likeliness that a certain constraint will be satisfied is larger than a predefined value. For this application knowing the quantile value can be more interesting than knowing the average or median value.
- We chose a kernel-based method because using non-linear kernels allows the learning of non-linear dependencies. Kernel-based methods in the batch setting, like SVM [12] and LS-SVM [13], have also shown to be successful for various applications such as optical character recognition and electricity load prediction.

Section 4.1 investigates the response time of some real-life web services and discusses the data used for our experiments. In Section 4.2 we give a short introduction to online learning with kernels and quantile estimation. In Section 4.3 two online quantile estimators are explained. Finally, a performance measure according to which different quantile estimators can be compared, is discussed in Section 4.4.

4.1 Quality of Web Service Data

A current problem for experiments regarding QoS of real web services is the lack of available datasets. For our evaluation, we found no usable time series on Quality of Web Service attributes. Service providers usually only publish average values for their services. The QWS dataset [2] of Al-Masri et al. [11] includes measurements of 9 QoS attributes for 2500 real web services. Each service was tested over a ten-minute period for three consecutive days. However, only the average QoS values are publicly available. Another public dataset is WS-DREAM [3] which offers real invocation info on 100 web services by using 150 distributed computer nodes located all over the world. The dataset contains data on consecutive invocations of the services but is limited to 100 time serie datapoints per service, which is not sufficient for our experiments. There is also no labeling on the time span of the different invocations of a service. Both datasets are restricted to short-running automated web services.

To cope with the data problem, we have collected real time series data for short-running online services ourselves. We used Web Inject [4], a free client-side monitoring tool for automated testing of web applications and web services. The tool allows to send soap requests to web services to analyze their response time and fault counts. We monitored a set of 8 popular online web services over a two-minute period for 7 consecutive days. Figure 4 illustrates the response time of a web service that allows a client to retrieve information on movies and theaters in the US. We can observe that a QoS attribute like response time can be very dynamic in time. Therefore, we believe that algorithms predicting QoS variations in time can contribute to existing literature on QoS-based web service composition.

[2]http: //www.uoguelph.ca/~qmahmoud/qws/index.html
[3]http://www.wsdream.net:8080/wsdream/
[4]http://www.webinject.org/

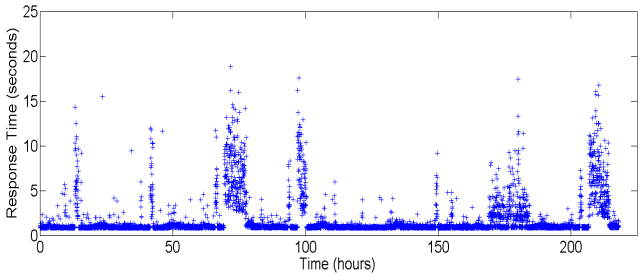


Figure 4: Time-varying RTs of online web service

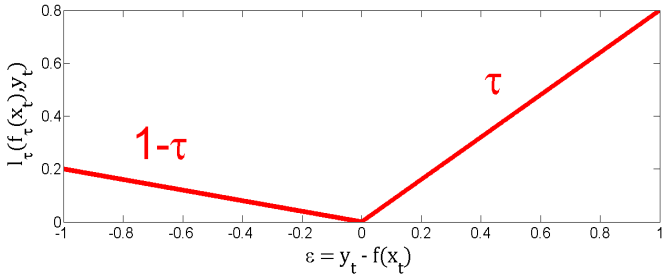


Figure 5: Pinball loss function used for quantile estimation. On the figure τ equals 0.8.

We believe that our approach can also contribute in the area of long-running workflow processes where human intervention might be required to fulfill a task. An example of such process is the mammography screening workflow described in Section 2. Also airline industries define business processes where a task consists of bringing passengers luggage to an airplane, technical checkup of a plane, etc. Precise predictions are then crucial to avoid delaying take-offs. For this kind a services, it is even more difficult to find real datasets. We evaluate them by means of simulated data in Section 5.

4.2 Kernel-Based Online Quantile Estimation

4.2.1 *Online Learning with Kernels*

Regression is a common task in machine learning. Given a training set $\mathcal{S}$ containing input vectors $\mathcal{X} = \{x_t\}_{t=1}^m$ along with corresponding output values $\mathcal{Y} = \{y_t\}_{t=1}^m$, the task is to find a function f that defines the relation between these input vectors and output values such that the errors on unseen data are as small as possible. We assume that the function f belongs to a Reproducing Kernel Hilbert Space (RKHS) with k the corresponding kernel function. The loss function $l(f(x_t), y_t)$ represents the cost for each mistake. Commonly used loss functions within kernel-based methods are the ϵ-insensitivity loss function [12] (1) and the quadratic loss function (2):

$$L_\epsilon(f(x_t), y_t) = \begin{cases} 0 & \text{if } |y_t - f(x_t)| \leq \epsilon \\ |y_t - f(x_t)| - \epsilon & \text{otherwise} \end{cases} \quad (1)$$

$$L_2(f(x_t), y_t) = (y_t - f(x_t))^2. \quad (2)$$

A measure of quality for f given the set of training examples $\mathcal{S}$ is the regularized risk

$$R_{reg}[f, S] = \frac{1}{m}\sum_{t=1}^{m} l(f(x_t), y_t) + \frac{\lambda}{2}\|f\|_H^2$$

where the inner product $\|f\|_H = < f, f >_h^{1/2}$ induces a norm on all f belonging to the RKHS and λ equals the regularization parameter. Unlike batch learning methods, most online learning methods handle only one example at a time. Minimizing a risk defined over multiple examples is not desirable in this setting. For that reason Kivinen et al. [14] proposed a method called NORMA that uses the instantaneous risk which only depends on one training example at a time and is defined as

$$R_{ins}[f, (x_t, y_t)] = l(f(x_t), y_t) + \frac{\lambda}{2}\|f\|_H^2.$$

At each step NORMA performs gradient descent with respect to the instantaneous risk. This results in the following update rule

$$f_{t+1} = (1 - \eta\lambda)f_t - \eta\frac{\partial l(f_t(x_t), y_t)}{\partial f_t}k(x_t, .) \quad (3)$$

where k is a kernel and η is the learning rate. The parameter λ allows truncation.

4.2.2 *Quantile Estimation*

An important goal of our application is to avoid the SLAs to be violated. In a lot of cases one can never be completely sure that all constraints will be satisfied. It would be nice, however, if we could select all services that satisfy a constraint with a chance predicted to be greater than a predefined value. Suppose, for example, we want to select all services that have a chance of at least τ to have a response time smaller than f_{max}. This can be achieved by first obtaining the confidence interval $[0, f_\tau(x_t)]$ such that the chance the response time belongs to this interval equals τ. The service is then selected if $f_\tau(x_t) \leq f_{max}$. The true quantile value is denoted as μ_τ.

We will obtain the function f_τ by using the pinball loss function [15] (Figure 5):

$$l_\tau(f_\tau(x_t), y_t) = \begin{cases} \tau(y_t - f_\tau(x_t)) & y_t - f_\tau(x_t) \geq 0 \\ (\tau - 1)(y_t - f_\tau(x_t)) & y_t - f_\tau(x_t) < 0. \end{cases}$$

Applying quantile estimation with the above loss function has some important benefits:

- The estimation is distribution-free in the sense that no assumption on the conditional distribution of the output value given an input vector needs to be made.
- The estimator is robust and thus resistant to outliers. Its breakdown point equals $1 - \tau$ if $\tau \geq 0.5$.
- The estimator ensures, in the batch setting, the quantile curve divides the observations in the desired ratios (τ and $1 - \tau$).

It has important disadvantages as well:

- Different quantile curves can cross each other because each quantile function is estimated independently.
- The estimator is very sensitive to changes in the output values of datapoints near the quantile curves.
- As for most, if not all, quantile estimators we cannot assure that out-of-sample observations are divided in the desired ratios.

Implementing this loss function in (3) results in the following update rule:

$$f_{\tau,t+1} = \begin{cases} (1-\eta\lambda)f_{\tau,t} + \eta\tau k(x_t,.) & y_t \geq f_{\tau,t}(x_t) \\ (1-\eta\lambda)f_{\tau,t} - \eta(1-\tau)k(x_t,.) & y_t < f_{\tau,t}(x_t). \end{cases}$$

In the batch setting, the combination of quantile estimation and kernels is already done by Takeuchi et al. [16].

4.3 Proposed Algorithms

4.3.1 One-Step-Ahead Quantile Estimation

The first algorithm we discuss is an algorithm that predicts one-step-ahead and has no input. The learned function is a constant function that changes each step. Because no truncation is needed for this method - the learned constant is the only value that needs to be memorized - we set λ equal to zero. The algorithm has the following update rule:

$$f_{\tau,t+1} = \begin{cases} (f_{\tau,t} + \eta\tau, y_t) & y_t \geq f_{\tau,t} \\ (f_{\tau,t} - \eta(1-\tau), y_t) & y_t < f_{\tau,t}. \end{cases} \quad (4)$$

This online learning algorithm is not kernel-based because there is no input.

We recommend using this algorithm when:

- Quantile values for the immediate future need to be predicted. The algorithm is designed to predict the next quantile value. This quantile value might not be representative for tasks that need to be completed in the distant future.
- Recent datapoints have the greatest predictive power. Datapoints from the distant past have less influence than recent datapoints. In case of seasonality, for example, other methods are recommended.
- The response time (RT) does not exceed the period during which a training point has predictive power. The RT at time t is only known at time $t + RT$.

4.3.2 Conditional Quantile Estimaton

The second algorithm we discuss is an algorithm that predicts the output (f.e. the response time), given a specific time as input (f.e. the time within a day). This estimator can predict multiple steps ahead. Using as input the time for which you require the response time, the estimator outputs the estimated quantile value for that time. For a monitor that monitors every two minutes and an input representing the time within a day, the maximum number of support vectors equals 720 (24 hours/day times 60 minutes/hour divided by 2 minutes). No truncation is needed if 720 is an acceptable number of support vectors and λ can be set to zero. The function $f_{\tau,t}$ can be written as a kernel expansion

$$f_{\tau,t}(x) = \sum_{i=0}^{719} \alpha_{i,t} k(x_i, x).$$

Initially all support values are set to zero ($\{\alpha_{i,0} = 0\}_{i=0}^{719}$). The algorithm has the following update rule:

$$i = t \mod 720$$

$$\alpha_{i,t+1} = \alpha_{i,t} + \begin{cases} \eta\tau & y_t \geq f_{\tau,t}(x_t) \\ \eta(\tau-1) & y_t < f_{\tau,t}(x_t). \end{cases}$$

The relation between the response time and time within a day is most likely non-linear, which justifies the use of kernels. Instead of using the time within a day, one can use the time within a week to get a different prediction for each week day.

We recommend using this algorithm when:

- Predictions further into the future need to be made. This algorithm can predict for any time.
- Similar patterns reoccur at multiples of the time range.

4.3.3 Adaptive learning rate

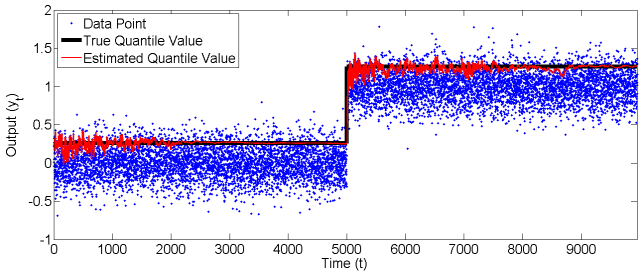
Instead of using a predefined learning rate, we can adapt the learning rate based on the predictions. If the predictions show that the learning rate is too high, then we lower the learning rate and vice versa. The adaptive learning rate discussed in this section is developed for the algorithm explained in Section 4.3.1. We denote $\frac{\partial l(y_t,f_t)}{\partial f_t}$ as $l'(y_t, f_t)$.

A too high learning rate causes the estimate to move past the expected optimum such that the expected value of the next change in the estimate is of opposite sign. A too low learning rate, on the other hand, causes the estimate to move to little in the direction of the expected optimum such that the expected value of the next change in the estimate has the same sign. The change in the estimate is proportional to the derivative of the loss. This means a too high learning rate causes $l'(y_t, f_t)$ and $l'(y_{t+1}, f_{t+1})$ to be negatively correlated, while a too high learning rate causes $l'(y_t, f_t)$ and $l'(y_{t+1}, f_{t+1})$ to be positively correlated.

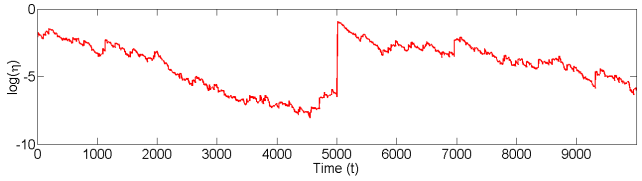
We propose the following update rule for the learning rate

$$\eta_{t+1} = \eta_t \exp\left(al'(f_{t-1}(x_t), y_t)l'(f_t(x_t), y_t)\right) \quad (5)$$

with a the parameter that determines the magnitude of the adjustments. This update rule will, on average, increase the logarithm of learning rate when the learning rate is too low and decrease the logarithm of learning rate when the learning rate is too high.



(a) 90%-quantile estimation



(b) Learning Rate

Figure 6: 99%-quantile estimation and corresponding learning rate using One-Step-Ahead prediction with adaptive learning rate.

For the pinball loss function, (5) becomes

$$\eta_{t+1} = \begin{cases} \eta_t \exp\left(a\tau^2\right) & y_t > f_{\tau,t}(x_t), y_{t-1} > f_{\tau,t-1}(x_t) \\ \eta_t \exp\left(a\tau(\tau-1)\right) & y_t < f_{\tau,t}(x_t), y_{t-1} > f_{\tau,t-1}(x_t) \\ \eta_t \exp\left(a\tau(\tau-1)\right) & y_t > f_{\tau,t}(x_t), y_{t-1} < f_{\tau,t-1}(x_t) \\ \eta_t \exp\left(a(\tau-1)^2\right) & y_t < f_{\tau,t}(x_t), y_{t-1} < f_{\tau,t-1}(x_t). \end{cases}$$

Figure 6 shows how the learning rate changes on a toy example. As long as the underlying distribution of the output value remains the same, the learning rate decreases such that the accuracy of the estimator can increase. The moment the underlying distribution changes, the learning rate increases such that the estimator can adapt fast to the new quantile value.

4.3.4 *Crossing Quantile Curves*

Crossing quantile curves can be an embarrassing phenomenon [16]. In this section we will propose a new update rule for the algorithm discussed in 4.3.1. It can be proven by induction this new update rule avoids quantile crossings when using the same fixed learning rate η for each quantile curve. The new update rule is chosen as follows:

$$f_{\tau,t+1} = \begin{cases} \min\left(f_{\tau,t} + \eta\tau, y_t\right) & y_t \geq f_{\tau_1,t} \\ \max\left(f_{\tau,t} - \eta(1-\tau), y_t\right) & y_t < f_{\tau_1,t}. \end{cases} \quad (6)$$

Suppose we want to avoid crossings between quantiles τ_1 and τ_2. We can assume, without further implications, $\tau_2 > \tau_1$. At the start of the training phase, we set $f_{\tau_2,1}$ larger than or equal to $f_{\tau_1,1}$. It can be proven that the inductive rule

$$f_{\tau_2,t+1} \geq f_{\tau_1,t+1} \text{ if } f_{\tau_2,t} \geq f_{\tau_1,t}$$

holds for (6). Due to space limitations, we omit this proof.

The new update rule makes, in our opinion, more sense because $f_{\tau,t+1}$ should be a compromise between $f_{\tau,t}$ and y_t, which implies $f_{\tau,t+1}$ should be a value between $f_{\tau,t}$ and y_t. Increasing $f_{\tau,t}$ should also never decrease $f_{\tau,t+1}$, which can occur when using the old update rule (4). For the conditional quantile estimator explained in Section 4.3.4 the quantile curves can still cross. How to avoid quantile crossings in a conditional online setting is an interesting topic for future research.

4.4 Performance Measure

In order to compare different quantile estimators we need a performance measure. The most straight-forward performance measure is how good the quantile property holds:

$$C_1 = |p(y_t > f_\tau(x_t)) - \tau|.$$

A constant function ($f_{\tau,t}(x) = c$) can satisfy the quantile property perfectly while being less good than some time-varying estimators. That's why we use another performance measure, the average loss:

$$C_2 = \frac{1}{N}\sum_{t=1}^{N} l\left(f_{\tau,t}(x_t), y_t\right). \quad (7)$$

The lower the average loss, the better the estimator.

5. EVALUATION

The algorithm explained in Section 4.3.1 will be evaluated using a real-life data set: the Movie Dataset discussed in Section 4.1. We will compare a quantile estimator with adaptive learning rate, a quantile estimator with fixed learning rate and an average estimator with adaptive learning rate. The average estimator differs from the quantile estimator in the sense that it minimizes the quadratic loss function (2). The hyperparameters are optimized according to (7) using the first 1500 data points. The algorithms are evaluated on the

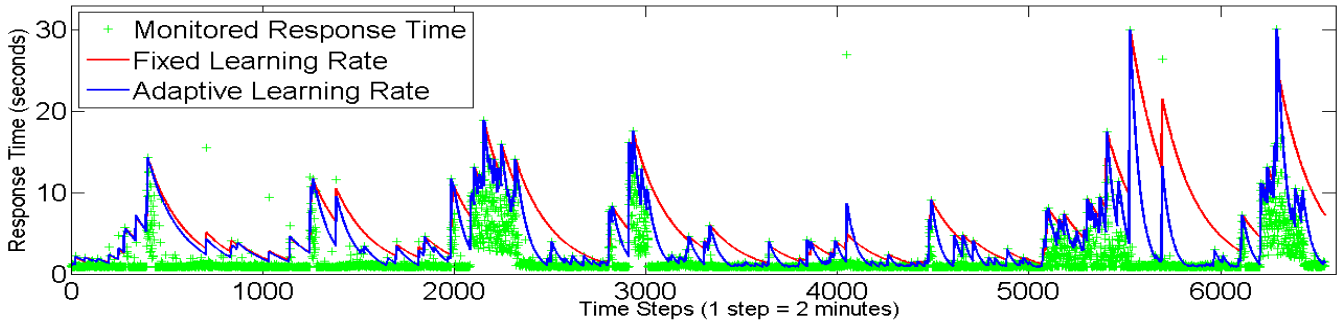


Figure 7: 99%-quantile estimation on the movie dataset using the One-Step-Ahead predictor with fixed and adaptive learning rate.

	99%-Quantile Estimation Adaptive Learning Rate		99%-Quantile Estimation Fixed Learning Rate		Average Estimation Adaptive Learning Rate		
Maximal RT	Acceptance Rate (%)	Violation Rate (%)	Acceptance Rate (%)	Violation Rate (%)	Acceptance Rate (%)	Violation Rate (%)	Violation Rate without Rejection (%)
2	43.6	1.55	12.7	1.41	82.7	4.63	19.1
3	58.3	1.02	29.3	1.08	86.0	3.50	14.8
5	73.3	0.51	52.5	0.48	91.7	3.01	8.5
8	84.3	0.47	70.1	0.31	98.2	3.13	3.9

Table 3: Violation and acceptance rates using the one-step-ahead predictor on the Movie Dataset. 99%-Quantile estimation with adaptive learning rate, 99%-quantile estimation with fixed learning rate and average estimation with adaptive learning rate are compared.

remaining 5000 data points. A request is accepted if the estimated quantile value is lower than the maximal allowed response time (RT). The violation rate equals the fraction of accepted requests that exceed the allowed RT. The quantile estimations and corresponding learning rates are shown in Figure 7 and Figure 8 respectively. Table 3 summarizes the violation and acceptance rates of the estimators given different maximal allowed RTs. We can see that the violation rate can be improved significantly by rejecting only a small number of requests. For example, using the quantile estimator with adaptive learning rate and given a maximal RT of 5, the violation rate can be reduced from 8.5% to 0.51% by rejecting only 26.7% of the requests. The adaptive learning rate causes, on average, a slightly higher violation rate while having a significant higher acceptance rate compared to the fixed learning rate. Rejection based on the average value causes a higher violation rate and a higher acceptance rate. Average-based rejection does not attempt to keep the violation rate below a certain value and for that reason we recommend quantile-based rejection.

The algorithm explained in Section 4.3.4 will be evaluated using the simulated data for a long-running process shown in Figure 9. In this simulation we assume the number of working hours needed to complete a task is lognormal distributed. A working day starts at 8h and ends at 16h. Outside this range the progress of the task is freezed. For example, a task taking 9 working hours to complete that starts at 10h will end at 11h the next day. The response time of 25 hours consist of 6 working hours within the same day, 16 non-working hours and 3 working hours the next day. Monitoring is done every 10 minutes and starts at 8h on day 1. Because the response time can become two or more days, we cannot use the One-Step-Ahead predictor. We choose a radial basis function kernel ($k(x_i, x_j) = \exp(\|x_i - x_j\|^2/\sigma^2)$). The hyperparameters are optimized by minimizing (7) using the fourth day as a validation set. The optimal parameters are: $\sigma = 6$ and $\eta = 0.02$. On Figure 9 we can see there can be more than half a day difference between quantile values at different moments. For the scenario explained in Section 2, a screening task with similar behavior that needs to be performed in the evening can sometimes better be outsourced to a screening center at the other side of the globe because the working day just starts there and the response time might be half a day faster. Table 4 shows the violation rate and the acceptance rate for different maximal response times given the real probability densities. We can see the algorithm allows the violation rates to be reduced significantly by rejecting the requests on well chosen moments.

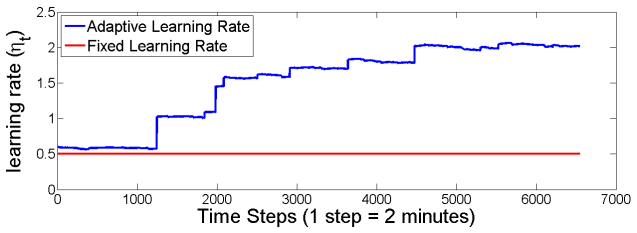


Figure 8: Learning rate of the 99%-quantile estimation on the Movie Dataset using the One-Step-Ahead predictor with fixed and adaptive learning rate.

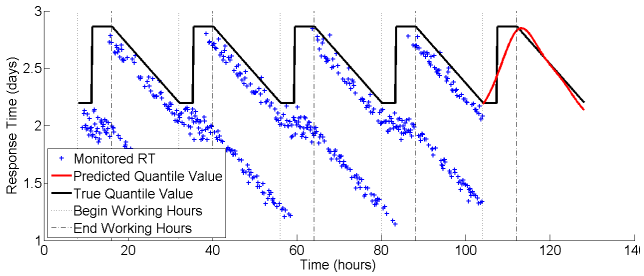


Figure 9: Estimating the 99%-quantile value of the response time on simulated data using the conditional quantile estimator.

Maximal RT	Accept. Rate (%)	Violation Rate (%)	Violation Rate without Rej. (%)
2.2	7.6	2.24	31.9
2.3	21.5	0.50	26.4
2.5	47.9	0.28	15.7

Table 4: Violation and acceptance rates using the conditional 99%-quantile estimator on the simulated data.

6. RELATED WORK

QoS in the context of web service composition has been widely discussed in several works [2-6,8-10]. Most research in this area focuses on determining QoS estimates for service compositions, given the QoS values of the atomic tasks. These solutions can be used to help service providers to find an optimal combination of web services from a pool of candidate services according to the QoS constraints. We elaborate on QoS-aware service composition in general, QoS-aware service composition in volatile environments and finally on Quality of web service prediction.

Cardoso's PhD. thesis [2] is a seminal work that proposes a framework that uses Stochastic Workflow Reduction for estimation of QoS in web service compositions. Canfora et al. [4] apply a similar model with minor adaptations. Their middleware uses Genetic Algorithms for deriving optimal QoS compositions. Hwang et. al. [8] propose a probabilistic framework for QoS computation. They extend Cardoso's model such that each QoS parameter for a web service is represented as a discrete random variable with a probability mass function. Zeng et al. [3] also present a QoS-aware middleware for quality-driven web service compositions. In this work, the authors propose state charts and aggregation functions to represent the execution plans and execution paths. Two service selection approaches for constructing composite services have been proposed: local optimization and global planning. Their study shows that global planning is better than local optimization. A practical approach is taken by Mukherjee [9]. They propose a model for estimating three key QoS parameters - Response Time, Cost and Reliability - of an executable BPEL process from the QoS information of its partner services and certain control flow parameters.

Most service composition models discussed above, however, are deterministic and thus require point estimates, such as expected values, for all quality measures. Other works take into account the fact that Business environments seldom remain unchanged over the lifetime of a Web process.

Harney [5] present a composition solution that intelligently adapts workflow processes to changes in quality parameters of service providers. Changes are introduced by means of expiration times, i.e. service providers provide their current reliability rates and duration of time for which the current reliability rates are guaranteed to remain unchanged. Wiesemann [6] formulate the service composition problem as a multi-objective stochastic program which simultaneously optimizes QoS parameters wich are modelled as decision-dependent random variables. Their approach accounts for quality uncertainty in a mathematically sound manner.

In this paper, we do not focus on optimization of the composition itself, but rather on how accurate expected values for quality measures of the constituent tasks can be achieved prior to execution. We observed that there is little related work that addresses QoS prediction for web services. Shao et al. [10] proposes predictive methods based on making similarity mining and prediction from consumers' experiences. Consumers that have similar historical QoS experiences on some services will likely have similar experiences on other services. They show that predicting QoS using their collaborative filtering based approach performs much better than average prediction. However, their goal is different from ours: they do not try to predict future expectations of QoS from a single point of view.

7. CONCLUSION

In this paper we propose a framework for QoS-aware web service composition. The framework is based on the Model-View-Controller (MVC) pattern, commonly used for adding dynamism to web pages. Based on predicted QoS attributes (Model), this framework can dynamically adapt a workflow instance. A workflow is designed as a master process which represents a template where tasks can be specified on an abstract level (View). Concrete implementations, modeled as aspects, are then selected by the adaptation logic according to the SLA (Controller). In contrast to most related work, we do not propose a solution for optimizing the composition process itself, but rather focus on how to achieve accurate estimates for the QoS values of its constituting tasks. We discuss two online quantile estimators that are used to minimize the violation rate of SLAs. The One-Step-Ahead predictor is not kernel-based and can only be used to predict quantile values for the immediate future given the response time is sufficiently low. The conditional quantile estimator is kernel-based and uses time as an input. It can be used when the response time is high or when quantile values further in the future need to be predicted. We have shown both estimators can improve service selection by drastically reducing the violation rate of a SLA with respect to minimizing the rejection rate of candidate compositions. Both algorithms have a low computational complexity (linear in the number of training data points) and their models can easily be updated with each new training example.

8. ACKNOWLEDGEMENTS

This work was supported by IBBT, the Research Council K.U.Leuven (GOA Ambiorics, GOA MaNet, CoE EF/05/006 Optimization in Engineering(OPTEC), IOF-SCORES4CHEM, several PhD/postdoc & fellow grants), the Flemish Government (FWO: PhD/postdoc grants, projects: G0226.06 (cooperative systems and optimization), G0321.06 (Tensors), G.0302.07 (SVM/Kernel), G.0320.08 (convex MPC), G.0558.08 (Robust MHE), G.0557.08 (Glycemia2), G.0588.09 (Brain-machine) research communities (ICCoS, ANMMM, MLDM); G.0377.09 (Mechatronics MPC)), IWT (PhD Grants, Eureka-Flite+, SBO LeCoPro, SBO Climaqs, SBO POM, O&O-Dsquare), the Belgian Federal Science Policy Once (IUAP P6/04 (DYSCO, Dynamical systems, control and optimization, 2007-2011)), EU (ERNSI; FP7-HD-MPC (INFSO-ICT-223854), COST intelliCIS, FP7-EMBOCON (ICT-248940)), Contract Research (AMINAL) and others (Helmholtz: viCERP, ACCM, Bauknecht, Hoerbiger).

9. ADDITIONAL AUTHORS

Sam Michiels (IBBT-DistriNet, Dept. Computer Science, K.U.Leuven, email: `sam.michiels@cs.kuleuven.be`).

10. REFERENCES

[1] M. Papazoglou and K. Pohl, "Report on Longer Term Research Challenges in Software and Services". Results from two workshops held at the European Commission premises at 8th of November 2007 and 28th and 29th of January 2008, with contributions from M. Boniface, S. Ceri, M. Hermenegildo, P. Inverardi, F. Leymann, N. Maiden, A. Metzger and T. Priol, European Commission, `http://www.cordis.lu`, 2008.

[2] J. Cardoso, *Quality of Service and Semantic Composition of Workflows*, PhD thesis, Univ. of Georgia, 2002.

[3] L. 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, no. 5, pp. 311-327, May 2004.

[4] G. Canfora, M. D. Penta, R. Esposito, and M. L. Villani,"An approach for QoS-aware service composition based on genetic algorithms," *Genetic and Evolutionary Computation Conference (GECCO 2005),* pp. 1069-1075, Washington DC, USA, 2005.

[5] J. Harney and P. Doshi. "Speeding up adaptation of web service compositions using expiration times," *International conference on World Wide Web (WWW 2007)*, pp. 1023-1032, New York, USA, ACM, 2007.

[6] W. Wiesemann, R. Hochreiter, and D. Kuhn, "A Stochastic Programming Approach for QoS-Aware Service Composition," *International Symposium on Cluster Computing and the Grid (CCGrid 2008)*, pp. 226-233, Lyon, France, May 2008.

[7] K. Geebelen, E. Kulikowski, E. Truyen and W. Joosen, "A MVC Framework for Policy-Based Adaptation of Workflow Processes: A Case Study on Confidentiality," *IEEE International Conference on Web Services (ICWS 2010)*, pp.401-408, 2010

[8] S.-Y. Hwang, H. Wang, J. Tang, J. Srivastava, "A probabilistic approach to modeling and estimating the QoS of web-services-based workflows," *Information Sciences*, vol. 177, no. 23, pp. 5484-5503, 2007.

[9] D. Mukherjee, P. Jalote, M. Nanda, "Determining QoS of WS-BPEL Compositions," *International Conference on Service-Oriented Computing (ICSOC 2008)*, pp. 378-393, Springer, Heidelberg, 2008.

[10] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized QoS Prediction for Web Services via Collaborative Filtering," *IEEE International Conference on Web Services (ICWS 2007),* pp. 439-446, 2007.

[11] E. Al-Masri and Q. H. Mahmoud, "Investigating web services on the world wide web," *International conference on the World Wide Web (WWW 2008)*, pp. 795-804, New York, USA, 2008.

[12] V. Vapnik, *Statistical learning theory*, New York: John Wiley & Sons, 1998.

[13] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor and J. Vandewalle, *Least Squares Support Vector Machines*, Singapore: World Scientific Pub. Co., 2002.

[14] J. Kivinen, A.J. Smola and R.C. Williamson, "Online learning with kernels," *IEEE Transactions on Signal Processing,* vol. 52, no. 8, pp. 2165-2176, 2002.

[15] R. Koenker and G. Bassett, "Regression quantiles," *Econometrica,* vol. 46, no. 1, pp 33-50, 1978.

[16] I. Takeuchi, Q.V. Le, T.D. Sears and A.J. Smola, "Nonparametric Quantile Estimation," *Journal of Machine Learning Research,* vol. 7, pp. 1231-1264, 2006.