Request Replication: An Alternative to QoS Aware Service Selection

Anis Yousefi and Douglas G. Down

Department of Computing and Software, McMaster University,
1280 Main St. W., Hamilton, Canada
{yousea2, downd}@mcmaster.ca

Abstract. QoS aware service selection involves finding services to match a client's functional and non-functional requirements. Current QoS advertisements typically provide a single value to represent the distribution of a non-functional property such as response time. However, the SOA literature implies that non-functional properties such as response time have inherently high variance in their values and thus representing non-functional properties with a single value does not reveal much about their actual distribution. This makes it hard for service clients to choose any selection strategy other than the conventional QoS aware service selection which is geared for clients choosing "a single service" to serve their needs. In this paper, we propose a new strategy for QoS aware service selection which takes advantage of the existing variability in QoS data to provide higher quality services with less cost compared to the conventional QoS aware service selection methods. In this method, we replicate each request over multiple independent services to achieve the required QoS. We also present a number of recommendations regarding the QoS advertisements in SOA so as to reveal more information about underlying distributions and thus enabling more sophisticated selection strategies. We will show using various examples how this approach works and enhances the conventional QoS aware service selection methods.

Keywords: SOA; QoS Advertisements; QoS Aware Service Selection; Request Replication;

1 Introduction

Service Oriented Architecture (SOA) is an increasing trend in developing business applications. In this architecture, software functionality is represented as a set of services with well defined interfaces which can be reused to build various types of applications. A service is published by a service provider (from now on, we will simply use "provider") and used by one or more service clients ("client" for short). Research in web services includes many challenging areas such as service composition, quality of service (QoS) aware service selection, etc.

QoS aware service selection is the process of choosing a service implementation from a pool of previously located services in a way that the selected service satisfies a set of QoS constraints. In SOA, there are two general approaches for

satisfying the QoS requirements. In the first approach, a client chooses from the pool of available services, the service with matching non-functional advertisement. An alternative to this approach is *QoS Negotiation*, in which the client negotiates with the provider to reach an agreement with regard to the non-functional requirements.

Non-functional properties of services, such as response time, are stochastic in nature. The dynamics of the environment in which a service is deployed, such as network-related delays and server congestion, can result in high variability in service non-functional properties. This yields two outcomes: On the negative side, the selected service may for a particular service invocation have response time that significantly exceeds the average value advertised. On the positive side, one could take advantage of the inherent variability in non-functional properties of a service to propose alternative service selection strategies.

In this paper, we present a novel alternative strategy, namely *Request Replication*, to satisfy the QoS requirements of a client in a more cost-efficient way by taking advantage of the existing high variance in non-functional properties. Unlike conventional QoS aware service selection, in *Request Replication* we choose from available services "a set" of independent low-cost and low-quality services in a way that their cooperation provides the required QoS. Throughout this work, we specifically consider response time as a representative of performance related non-functional properties. We concurrently send a request to a set of services, take the fastest response, and discard the remaining requests.

The term "replication" is also used in other contexts in the literature of QoS aware service selection. For example, *Service Replication* is a mechanism providers use to guarantee the quality of service that they have obliged to in their service level agreements (SLAs) [20]. Also, the idea of sending a request to multiple functionally equivalent services along with voting mechanisms are used in the context of fault tolerance [6, 9, 22].

In this paper we enhance the state of QoS aware service selection in SOA from the client's perspective. The contributions of the paper are:

1. We provide an alternative strategy to conventional "QoS aware service selection" in SOA, which has the potential to allow clients get better quality services with less cost. In this method, a client can build better quality services from low cost, low quality ones.
2. We present a number of recommendations about service advertisements for performance related non-functional properties, and specifically service response time. We believe that the advertisements should provide enough information about non-functional properties of a service to enable clients to make better decisions with regard to service selection. For this reason, we believe that clients should know the distribution of non-functional properties, or enough of the parameters of the distribution to construct estimates.

The rest of this paper is organized as follows: Section 2 explains the basics of QoS aware service selection and negotiation in SOA. Section 3 presents the proposed "Request Replication" strategy. Section 4 provides a number of recom-

mendations for service advertisements. Section 5 discusses related work in the literature of QoS aware service selection in SOA. Section 6 concludes the paper.

2 Service Selection in SOA

In this section, we discuss the QoS model of SOA. Specifically, we explain the basics of QoS-aware service selection and negotiation as well as the nature of QoS advertisements.

2.1 QoS-Aware Service Selection

Services in SOA are pieces of functionality, wrapped in well defined interfaces which are published for others to use. At the time of publication, a provider registers a service with a registry by providing information about the functionality and interface of the service along with its non-functional properties. QoS aware service selection is the process of choosing a service implementation from a pool of previously published services in such a way that the selected service satisfies a set of functional and non-functional requirements. The basic building blocks for any service selection approaches are discussed below [10].

Client Service Requirements
To find an appropriate service, a client submits to the service selection mediator a set of requirements along with their request. The requirements may involve both functional and non-functional aspects which need to be satisfied. Clients not only expect the service to meet functional aspects but they also require services to meet non-functional aspects that is, quality of services such as service performance, reliability, security, trust, execution cost, etc.

Provider Service Advertisements
The services offered by providers are concerned about functional and non-functional aspects. The providers thus specify both functional and non-functional properties of services in what is called a "service offering" or "service advertisement". The functional properties include service parameters, messages, behaviour and operation logic. The non-functional properties include QoS (security, reliability, response time, call cost, etc.) and Context (location, intention, client name, provider details, etc.). The non-functional properties are usually defined using a QoS ontology.

Service Selection Process
The service selection process involves finding a match for a client's requirements, among available service advertisements. In a basic form, service selection provides the best match for the client's requirements. In a more general form, service selection provides a ranking of the available services with regard to the client's requirements. Many service selection techniques and algorithms are proposed in the literature. These techniques can be divided into three categories [10]:

Functional Based service selection is concerned with retrieving functional descriptions from service repositories and examining them to see if they satisfy the functional requirements demanded by the client [1, 5].

Non-functional Based (or QoS aware) service selection is concerned with non-functional properties. With the rapidly growing number of available services, clients are presented with a choice of functionally similar services. This choice allows clients to select services that match other criteria, non-functional attributes, including QoS and context [14, 19].

User Based service selection involves the selection of best service among numerous discovered services based on the clients feedback, trust and reputation [11, 17].

2.2 QoS Negotiation

The requirements specified by a client may vary from the specified QoS in a service advertisement. In this case the provider and the client can enter a negotiation process to adjust the client's requirements and the provider's advertisement and reach an agreement accordingly [13, 16].

2.3 QoS Advertisements

So far there are no established standards for specifying QoS advertisements in a formal, machine-readable way [3]. However, various XML-based languages have been proposed such as Web Service Level Agreement (WSLA) [7] and Web Services Offerings Language (WSOL) [15]. Such languages help define contracts to specify agreed-upon, non-functional properties of Web services in the form of Service Level Objectives (SLOs). They also provide a model for measuring, evaluating, and managing the compliance with non-functional properties.

A service level objective expresses a commitment to maintain a particular state of the service in a given period. The state is defined as a logical expression over predicates that refer to non-functional properties and defines an obligation, that is, what is asserted by the provider to the client. An example of such an obligation is: "it is guaranteed that the average response time of the service is less than 5 seconds" [7].

Stantchev and Schropfer [12] recently proposed a structure for formalization of service level objectives and technical service capabilities. A service level objective in this structure has the following form: non-functional property + predicate + metric (value, unit) + percentage + if + qualifying conditions (non-functional property + predicate + metric). An example of such a SLO would be "The transaction rate of the service is higher than 90 transactions per second in 98% of the cases if throughput is higher than 500 kB/s."

3 The Request Replication Strategy

In this paper we are presenting an alternative strategy for QoS aware service selection. The proposed strategy benefits clients in potentially receiving better

services for less cost. In this method, a client uses multiple functionally equivalent services to get the quality they want while minimizing the service costs. We will show how and when using multiple services can increase the quality of service.

3.1 Motivating Example

Assume services S_1 to S_5 are different implementations of a calendar service with usage prices of \$40, \$10, \$20, \$10, and \$10 per month, respectively. Assume that the following service advertisements are provided by corresponding providers of S_1 to S_5.

S_1: The response time of S_1 is less than or equal to 9s in 96% of the cases
S_2: The response time of S_2 is less than or equal to 10s in 92% of the cases
S_3: The response time of S_3 is less than or equal to 10s in 92% of the cases
S_4: The response time of S_4 is less than or equal to 8s in 70% of the cases
S_5: The response time of S_5 is less than or equal to 8s in 70% of the cases

Assume that a client is looking for a calendar service with the following QoS requirement.

R: Response time of the service must be less than or equal to 9s in 96% of the cases

With this information, a conventional service selection mechanism will choose S_1, because it is the only service matching the QoS requirement. The price to be paid in this case is \$40 per month.

However, one can take a different strategy other than the conventional service selection. One could choose any combination of services, concurrently send a request to all of them, and pick the fastest response. In general, this new strategy could improve the results. Assuming that the service response times for S_1 to S_5 are mutually independent and exponentially distributed, we can show that choosing S_2 and S_3 would improve the results. In this case, the resulting response time would be less than 9s in 99% of the cases and the price to be paid would be \$30 per month. The details are as follows:

First, we represent the advertisements of S_2 and S_3 as

$$P(R_2 \leq 10s) = 0.92$$
$$P(R_3 \leq 10s) = 0.92.$$

Knowing that the distributions of R_2 and R_3 are exponential, in this step, we need to find the rate parameters λ_2 and λ_3 of the corresponding distributions. We have

$$P(R_2 \leq 10s) = 0.92$$
$$1 - e^{-10\lambda_2} = 0.92 \Rightarrow \lambda_2 = \frac{-ln0.08}{10} \approx 0.25.$$

Similarly,

$$\lambda_3 \approx 0.25.$$

Since we pick the fastest response, the resulting response time R_{min} will be equal to the minimum of R_2 and R_3 and thus it is exponentially distributed with parameter $\lambda_2 + \lambda_3$, that is

$$P(R_{min} \leq r) = 1 - e^{-(\lambda_2+\lambda_3)r}$$
$$P(R_{min} \leq 9) = 1 - e^{-(0.50)(9)} \approx 0.99.$$

This satisfies the client's requirement which is represented as

$$P(R_{req} \leq 9s) = 0.96.$$

Of course it is not at all clear that the exponential distribution is a reasonable choice for the individual response time distributions. However, if we have more information about the distribution of service response times we could estimate the underlying distributions. We will show in Section 3.4 how adding more information to current advertisements can change the results.

3.2 General Approach

In this paper, we are dealing with the general problem of QoS aware service selection, defined as:

Having functionally equivalent services S_1 to S_n, find the most cost efficient service(s) that match the QoS requirements of a client.

In other words, we need to minimize the costs of selected service while satisfying client defined constraints on non-functional properties.

We assume the following format for QoS advertisements.

$$P(R_i \leq r_i) \geq p_i$$
$$E[R_i] = m_i,$$

which is read as "the probability that the response time of service i is less than or equal to r_i is greater than or equal to p_i, where the mean response time of service i is m_i". We also assume the following format for a non-functional requirement.

$$P(R \leq r_{req}) \geq p_{req},$$

which is read as "the probability that the response time of the selected service(s) be less than or equal to r_{req} must be greater than or equal to p_{req}".

3.3 Request Replication

The proposed strategy for QoS aware service selection is called *Request Replication*. In this method, we choose one or more services whose aggregate QoS serves the needs of a client. We concurrently send a request to all the selected services, pick the fastest response and cancel the other requests. A key motivation comes from the idea that taking advantage of even a small amount of

additional choice for a client can lead to significant performance improvements. This idea has been explored by Mitzenmacher [8], amongst others, in another context (queuing problems).

Algorithm 1 presents a pseudo code description of the proposed *Request Replication* method. Similar to conventional QoS aware service selection, in the first step we need to find all services with matching functional properties. We call this set the *functionally eligible services (FES)* set. In the next step, we choose one or more services from FES whose aggregate QoS matches the request. This is done in two steps, as follows.

Algorithm 1 Request Replication

find all functionally eligible services and represent them as a set (*FES*).
fit appropriate distributions to the response time data of all functionally eligible services.
$cost = \infty$
for all $fes \subseteq FES$, $fes \neq \emptyset$ **do**
  if $(cost_{sum} = \sum_{s \in fes} s.cost) \leq cost$ **then**
    compute cumulative distribution function for the minimum response time distribution of the services in the subset *fes* at point r_{req}, that is
    $CDF_{min}(r_{req}) = 1 - \prod_{s \in fes} (1 - CDF_s(r_{req}))$,
    where $CDF_s(r_{req})$ is the cumulative distribution function for service s at point r_{req}.
    if $CDF_{min}(r_{req}) \geq p_{req}$ **then**
      $cost = cost_{sum}$
      $selectedServicesPool.replace(fes)$
    end if
  end if
end for

STEP 1 - Fit appropriate distributions to the response time data of all functionally eligible services.

In the first step, we need to find distributions that best describe the available service advertisements. The first question here is "what sort of distribution better fits the available service response time data in SOA?".

The choice of what distribution to fit and the method that we use to make the fit do not affect our approach and the insights provided in the rest of this section still hold. Gorbenko et al. [4] suggest that the Gamma distribution best describes response time data in SOA. Therefore, we represent response time data using Gamma distributions in this paper. As an illustration, we show how a Gamma distribution can be fit to a service advertisement of the form

$$P(R_i \leq r_i) \geq p_i$$
$$E[R_i] = m_i$$

The Gamma distribution is a two-parameter family of continuous probability distributions. It has a scale parameter θ and a shape parameter k. A random variable X that is Gamma-distributed with scale θ and shape k is denoted $X \sim Gamma(k, \theta)$. The mean and cumulative distribution function of the Gamma distribution can be expressed in terms of the Gamma function parametrized in terms of the shape parameter k and scale parameter θ. Both k and θ will be positive values. The mean of a Gamma-distributed random variable X is $k\theta$ and the cumulative distribution function of X is

$$CDF(x; k, \theta) = P(X \leq x) = \frac{\gamma(k, x/\theta)}{\Gamma(k)},$$

where $\gamma(k, x/\theta)$ is the lower incomplete Gamma function, defined as

$$\gamma(s, x) = \int_0^x t^{s-1} e^{-t}\, \mathrm{d}t,$$

and $\Gamma(k)$ is the Gamma function, defined as

$$\Gamma(k) = \int_0^\infty t^{k-1} e^{-t}\, \mathrm{d}t.$$

There is no fixed way to fit a Gamma distribution to an available advertisement. In this paper, we incorporate the *bisection search* method where we search for the root of the following function

$$f(k) = \frac{\gamma(k, r_i k/m_i)}{\Gamma(k)} - p_i$$

which is derived from the cumulative distribution function of the Gamma distribution (CDF) at $x = r_i$ where θ is replaced by m_i/k.

The bisection method searches for the root of $f(k)$ in an initial interval (a, b) such that $f(a)$ and $f(b)$ have opposite signs. Then, it iteratively divides up the interval in half in each step until it finds a sufficiently small interval that encloses the root. To find a and b we start by setting k to integer values and computing $f(k)$. Knowing that k is a positive value, we compute $f(k = iv)$ for $iv = 1, 2, ...$ until one of the following is true:

- $f(iv) = 0$: in this case the root has been found and is equal to iv.
- $f(iv).f(iv + 1) < 0$: in this case $a = iv$ and $b = iv + 1$.

The method now divides the interval (a, b) in two by computing the midpoint $c = (a + b)/2$ of the interval. Unless c is itself a root, there are now two possibilities: either $f(a).f(c) < 0$ in which case we select the interval (a, c), or $f(c).f(b) < 0$ in which case we select the interval (c, b) to continue.

As an example, assume that in the advertisement above, $r_i = 100$, $p_i = 0.81$ and $m_i = 66$. In the initial step, we find out that $k = 2$ results in $CDF(100) - 0.81 = -0.00466777$ and $k = 3$ results in $CDF(100) - 0.81 = 0.02147040$. In the next steps we continue breaking this interval in half and testing $CDF(100) - 0.81$ for $k = 2.5, 2.25, ...$ We find that $k = 2.17116$ is a very close fit. In fact, with 10^{-7} precision, the result is 0:

$$\frac{\gamma(2.17116, 217.116/66)}{\Gamma(2.17116)} - 0.81 = 0.00000003$$

STEP 2 - Check if any single service or combination of services satisfies the QoS requirements.

In this step, we need to find one or more services where the distribution of the minimum of their response times matches the requirements. We also need to select from available candidates, the set of services with minimum cost.

For this purpose, we first choose a subset of functionally equivalent services *FES* where the cumulative cost of services is less than the current cost, initially set to infinity. Then we compute the distribution of the minimum response times for services in the selected subset. The reason for that is, in *Request Replication* we choose the fastest response and thus the distribution of response times for the super service we are building is the equal to the distribution of the minimum response time of its underlying services. The cumulative distribution function for the minimum response time for a set of services fes is computed as

$$\begin{aligned} CDF_{min}(R_{min} = r) &= P(R_{min} \leq r) \\ &= P(Min_{s \in fes}(R_s) \leq r) \\ &= 1 - P(Min_{s \in fes}(R_s) > r) \\ &= 1 - P(\wedge_{s \in fes}(R_s > r)). \end{aligned}$$

It is generally hard to calculate this formula unless the response time of services in fes are mutually independent. In this case:

$$\begin{aligned} 1 - P(\wedge_{s \in fes}(R_s > r)) &= 1 - \prod_{s \in fes} P(R_s > r) \\ &= 1 - \prod_{s \in fes} (1 - \ CDF_s(R_s = r)). \end{aligned}$$

At this point we compute $CDF_{min}(r_{req})$ for all eligible subsets and update the pool of selected services, if a subset satisfies the requirement (i.e., $CDF_{min}(r_{req}) \geq p_{req}$).

3.4 Motivating Example Revisited

Adding to the previous example in Section 3.1, assume that we also know the means of the distributions.

$m_1 = 8, m_2 = 8, m_3 = 8, m_4 = 7.2, m_5 = 7.2$

Following Algorithm 1, in the first step we find matching Gamma distributions for all service advertisements. In this case:

$k_1 \approx 0.01, \theta_1 \approx 800$
$k_2 \approx 34, \theta_2 \approx 0.24$

$k_3 \approx 34, \theta_3 \approx 0.24$
$k_4 \approx 18, \theta_4 \approx 0.4$
$k_5 \approx 18, \theta_5 \approx 0.4$

In the next step we try different subsets of services, starting from single services, the results are:

S_1 is still a match.
S_2 or S_3 does not match ($CDF_2(9) \approx 0.78$ and $CDF_3(9) \approx 0.78$).
S_{2+3} (replication over S_2 and S_3) does not match ($CDF_{min}(9) \approx 0.95$).
S_4 or S_5 does not match ($CDF_4(9) \approx 0.86$ and $CDF_5(9) \approx 0.86$).
S_{4+5} (replication over S_4 and S_5) matches ($CDF_{min}(9) \approx 0.98$).

In this case, replication over S_4 and S_5 is preferred since it provides the required QoS with lower price of \$20. Note that the previous choice of S_2 and S_3 for replication is no longer an option. The extra "mean" information results in the construction of more accurate distributions for S_2 and S_3 which consequently indicates that replication over S_2 and S_3 does not actually satisfy the requirement.

3.5 Discussion

In this section we discuss a number of issues related to the *Request Replication* approach.

Independent Services. The underlying mathematics works well if service response times are mutually independent. In other words, for any two services S_i and S_j, $P(Min(R_i, R_j) > r) = P(R_i > r)P(R_j > r)$ iff R_i and R_j are independent. If response times are dependent, the calculation is not as straightforward. In this case, we would need information about each probability as well as the joint probability which may be difficult to calculate. Note that our approach should also work well if there is approximate independence, i.e. if the relation above approximately holds.

High Variance. The *Request Replication* method takes advantage of high variability in the response time data. If the response times were deterministic, the result of choosing more than one service would be the same as the result of choosing the service with minimum response time (which is what QoS aware service selection does). High variability in response times increases the chance that multiple services may complement each other with respect to performance and thus getting better response times via *Request Replication* is more likely. Therefore, the provided method improves the results over the conventional service selection as long as the response time data are highly variable. The type of distribution is not important.

Replication Overhead. Similar to other domains such as fault tolerance, replication in this work also introduces a number of overheads. Using *Request Replication*, one should think of a mediator which replicates a service call to selected services, returns the first response to the client and cancels the remaining calls. As suggested by [6] the replication overhead is small and thus acceptable.

4 Revisiting Service Advertisements

Looking at the literature, current QoS advertisements typically provide a single value (e.g., the average) to represent the response time of a service. For example, "Average response time of operation X from service Y is less than or equal to 0.5 seconds" or "Response time of operation X from service Y is less than or equal to 0.5 seconds in at least 95% of the cases". This makes it difficult for clients to take a different strategy other than the conventional service selection where the client is limited to choosing "a particular service" as the "best available choice" with regard to their needs.

The conventional service selection would work well if the non-functional properties of services were actually deterministic. However, the literature [4, 23] suggests that non-functional properties of services and in particular, service response time, are non-deterministic and highly variable due to dynamics of the environment in which services are deployed. For example, factors such as network traffic and server congestion greatly influence the response time of services and current advertisements do not reflect this high variance.

Knowing more about the actual distribution of non-functional properties provides more opportunities for clients, including:

Adjusting the requirements. Non-functional requirements are usually soft constraints. Clients may be willing to change their non-functional requirements based on the availabilities and the offered prices, as in the case of QoS negotiation. Providing more information about the actual distribution of non-functional properties makes it easier for the clients to make better decisions about their requirements. As an example, assume the following situation. Services S_1 and S_2 are advertised with average response times of 0.95 and 1.05 seconds, respectively. A potential client needs a service with an average response time of less than 1 second. With this information, the client would choose S_1 as the better choice. On the other hand, assume that we know variances of the response times for services S_1 and S_2. In this case, S_1 has high variance ($= 1$) and S_2 has low variance ($= 0$). Knowing this information, the client may be willing to revise their request and choose S_2 as the better choice as it is more reliable with regard to timing constraints.

Changing the selection strategy. As mentioned before, having a better understanding of the distribution of non-functional values, makes it possible for clients to choose from a variety of strategies for QoS aware service selection.

Request Replication is one example of such strategies which benefits clients by providing them with better quality services with less cost. There may also be other possibilities.

For these reasons we recommend providers to advertise the response time of their services using more than one representative values. This makes it possible to estimate a Gamma distribution for the response time of a service, which is shown to be a good fit for describing the response time distribution in SOA [4]. Providers can use any two pieces of information about the response time to advertise it. In this work we are considering the advertisements of the form

$$P(R \leq limit) \geq percentage$$
$$E[R] = m,$$

as this adds a lightweight change to the advertisements currently available. But, one could think of any other pieces of information, such as mean m and variance σ^2, for an advertisement.

No matter how the services are advertised, the *Request Replication* strategy can still be applied. As long as we estimate a distribution to the available advertisement we can use Algorithm 1 to find appropriate services for replication. If an advertisement provides one piece of information about the response time, we could fit an exponential distribution (a simple form of Gamma distribution, where $k = 1$) to it, and if two pieces of information are provided a Gamma fit could be estimated. Our algorithm, thus, is able to deal with a mix of advertisement formats which is very likely in a heterogeneous SOA environment.

5 Related Work

Although there are no established standards to specify QoS advertisements, many languages and notations have been proposed to capture QoS properties of a service [2, 7, 15, 18]. QoS languages provide for each service a number of obligations, each a one-point description of a non-functional property. We are recommending that providers advertise the non-deterministic non-functional values such as response time with at least two pieces of information so that more sophisticated decisions are possible with regard to what services best match clients' needs.

Current QoS service selection involves finding services that match a set of QoS requirements. Matching is a one to one comparison of an obligation and a requirement. Then, there is an aggregation process to evaluate a service with regard to all requirements. With this regard, the QoS aware literature on SOA basically considers the challenges of "semantic" matching, where semantic defines the relationship between QoS-related terms [2]. On the contrary, in this paper we are interested in finding a better matching based on the actual distribution of non-functional properties.

So far, we have not seen any approaches similar to ours in the literature of QoS aware service selection. The main QoS related tracks of research in SOA

use and extend the simple one-to-one matching with: the use of optimization algorithms to choose the "best" available service or rank services with regard to an application-specific utility function [18, 21], QoS negotiation [13, 16], and breaking down and dealing with global and local QoS constraints in composite services [19].

6 Conclusion

In this paper, we presented a novel alternative strategy to satisfy the QoS requirements of a client in a more cost efficient manner by taking advantage of the existing high variance in the real values of non-functional properties. In this method, we use multiple independent low-cost, low-quality services to satisfy the QoS requirements of a single request.

One of the advantages of our algorithm is that, no matter what format the advertisements have and how one estimates a distribution for an advertisement, one is still able to use the *Request Replication* algorithm and the underlying mathematics are valid. In other words, our proposed algorithm can deal with a mix of advertisement formats that providers may use in a SOA-based environment.

There is a possibility to combine the *Request Replication* idea with the QoS negotiation process already existing in the SOA literature. In this approach one can negotiate with the provider of a service to acquire a reasonably low price for a service (and of course loose QoS as a result) and then incorporate multiple services of this sort to acquire the required QoS. This idea is currently being investigated.

Also, the underlying assumption of independence should be considered more comprehensively. If service response times are correlated as a result of services sharing resources (e.g., shared services, network access, server, etc), the calculation of the minimum response time distribution becomes more difficult. As future work we are investigating how we can exploit knowledge about structure of the correlations to preform the required calculations.

References

1. Baldoni, M., Baroglio, C., Martelli, A. and Patti, V.: Reasoning about Interaction Protocols for Customizing Web Service Selection and Composition. J. Logic and Algebraic Programming. Elsevier. 70(1), 53–73 (2006)
2. Chaari, S., Badr, Y., Biennier, F.: Enhancing Web Service Selection by QoS-Based Ontology and WS-Policy. In: ACM symposium on Applied computing (SAC), pp. 2426–2431 (2008)
3. Chatterjee, A. M., Chaudhari, A. P., Das, A. S., Dias, T., Erradi, A.: Differential QoS Support in Web Services Management. http://soa.sys-con.com/node/121946
4. Gorbenko, A., Kharchenko, V., Mamutov, S., Tarasyuk, O., Chen, Y., Romanovsky, A.: Real Distribution of Response Time Instability in Service-Oriented Architecture. Technical Report, Newcastle University, UK (2009)
5. Klusch, M., Kapahnke, P.: Semantic Web Service Selection with SAWSDL-MX. In: International Semantic Web Conference, pp. 3–16 (2008)

6. Looker, N., Munro, M., Xu, J.: Increasing Web Service Dependability Through Consensus Voting. In: 29th Annual International Conference on Computer Software and Applications Conference (COMPSAC), pp. 66–69. IEEE Computer Society, Washington, DC (2005)
7. Ludwig, H., Keller, A., Dan, A., King, R. P., Franck, R.: Web Service Level Agreement (WSLA) Language Specification. IBM Corporation (2003)
8. Mitzenmacher, M.: The Power of Two Choices in Randomized Load Balancing. J. IEEE Transactions on Parallel and Distributed Systems. 12(10), 1094–1104 (2001)
9. Salatge, N., Fabre J.: Fault Tolerance Connectors for Unreliable Web Services. In: International Conference on Dependable Systems and Networks (DSN), pp. 51–60. IEEE Computer Society, Washington, DC (2007)
10. Sathya, M., Swarnamugi, M., Dhavachelvan, P., Sureshkumar, G.: Evaluation of QoS Based Web Service Selection Techniques for Service Composition. Intl. J. Software Engineering (IJSE). 1(5), 73–90 (2010)
11. Srivastava, A. , Sorenson, P. G.: Service Selection Based on Customer Rating of Quality of Service Attributes. In: IEEE International Conference on Web Services (ICWS). pp. 1–8 (2010)
12. Stantchev, V., Schropfer, C.: Negotiating and Enforcing QoS and SLAs in Grid and Cloud Computing. In: 4th International Conference on Advances in Grid and Pervasive Computing, pp. 25–35 (2009)
13. Swarnamugi, M., Sathya, M., Dhavachelvan, P.: A Negotiation Model for Web Service Selection. In: International Conference on Recent Trends in Soft Computing and Information Technology, pp. 251–256 (2010)
14. Tian, M., Gramm, A., Ritter, H., Schiller, J. H.: Efficient Selection and Monitoring of QoS-aware Web Services with the WS-QoS Framework. In: IEEE/WIC/ACM International Conference on Web Intelligence (WI), pp. 152–158 (2004).
15. Tosic, V., Patel, K., Pagurek, B.: WSOL Web Service Offerings Language. In: Ch. Bussler et al. (eds.) WES 2002. LNCS, vol. 2512, pp. 57-67. Springer-Verlag Berlin Heidelberg (2002)
16. Wang, Y., Wang, L., HuA, C.: QoS Negotiation Protocol for Grid Workflow. In: 5th International Conference on Grid and Cooperative Computing (GCC), pp. 195–198 (2006)
17. Wang, P., Chao, K. M., Lo, C. C., Farmer, R., Kuo, P. T.: A Reputation-Based Service Selection Scheme. In: IEEE International Conference on e-Business Engineering (ICEBE), pp. 501–506 (2009)
18. Yan, J., Piao, J.: Towards QoS-Based Web Services Discovery. In: G. Feuerlicht and W. Lamersdorf (eds.) ICSOC 2008. LNCS, vol. 5472, pp. 200-210, Springer-Verlag Berlin Heidelberg (2009)
19. Yang, Y., Tang, S., Xu, Y., Zhang, W., Fang, L.: An Approach to QoS-Aware Service Selection in Dynamic Web Service Composition. In: 3rd IEEE International Conference on Networking and Services (ICNS), pp. 18–23. IEEE Press, New York (2007)
20. You, K., Qian, Z., Tang, B., Lu, S., Chen, D.: QoS-Aware Replication in Service Composition. Int. J. Software and Informatics. 3(4), 465–482 (2009)
21. Yu, T., Lin, K. J.: Service Selection Algorithms for Composing Complex Services with Multiple QoS Constraints. In: 3rd International Conference on Service Oriented Computing, pp. 130–143 (2005)
22. Zheng, Z., Lyu, M.: A QoS-Aware Fault Tolerant Middleware for Dependable Service Composition. In: International Conference on Dependable Systems and Networks (DSN), pp. 239–248, IEEE (2009)
23. Zhu, L., Gorton, I., Liu, Y., Bui, N. B.: Model Driven Benchmark Generation for Web Services. In: International Workshop on Service Oriented Software Engineering (SOSE), pp. 33–39 (2006)