QoS monitoring of soft contracts for transaction based Web services orchestrations*

Albert Benveniste[1], Stefan Haar[3], Claude Jard[2], and Sidney Rosario[1]

[1] Irisa/Inria, Campus de Beaulieu, 35042 Rennes cedex, France
[2] Irisa/ENS Cachan, Campus de Beaulieu, 35042 Rennes cedex, France
[3] Irisa/Inria Rennes and SITE, University of Ottawa, Canada

Abstract. Web services orchestrations and choreographies require establishing Quality of Service (QoS) contracts with the user. This is achieved by performing QoS composition, based on contracts established between the orchestrator and the Web services called in the orchestration. Typical QoS parameters include maximum query throughput, response time, security, and validity of the response. Usually, QoS contracts are stated in the form of hard guarantees (e.g., response time always less than 5 msec). However, experiments and measurements from existing Web services show evidence that soft guarantees, not hard, should be stated instead (e.g., response time less than 5 msec in 95% of the cases). In a previous work [1] we have proposed using *soft contracts,* by taking a *probabilistic* approach. Contracts are characterized by means of a set of selected quantiles of probability distributions for QoS parameters. We showed how to compose such contracts, to yield a global QoS (probabilistic) contract for the orchestration. Our approach is supported by the *TOrQuE* tool, that performs probabilistic contract composition, automatically, from orchestration specification and QoS contracts with the Web services called by the orchestration.
In this work in progress, we develop monitoring techniques for detecting the violation of such soft probabilistic contracts. Monitoring of the called services is performed by the orchestration. We propose two simple statistical methods for monitoring the called services for contract violation.

1 Introduction

Web Services Orchestrations have attracted growing interest over the last years [7,20]. They are now considered an infrastructure of choice for managing business processes and workflow activities over the Web infrastructure [3]. In this context, the Web services for composition are mainly of transactional nature. BPEL [7] has become the industrial standard for specifying orchestrations. Numerous studies have been devoted to relating BPEL to mathematical formalisms for workflows, such as WorkFlow nets (WFnets) [8] a special subclass of Petri nets, or the pi-calculus [15]. This has allowed developing analysis techniques and

* This work is still in progress

tools for BPEL [16,18] including functional aspects of contracts [10], as well as techniques for workflow mining from logs [17].

When applied to the management of OEM/supplier cooperations, orchestrations must make precise the duties and responsibilities of the different actors in such chains. This is achieved by relying on *contracts* [6]. Contracts specify the requirements each subcontractor must satisfy; from these, the overall contract between orchestration and its customers can be established. This process is called *contract composition.*

While functional aspects of contract composition rely on the above mentioned formal models and techniques [10], *Quality of Service (QoS)* aspects must be handled as well. The Web Service Level Agreement (WSLA) framework [19] is a standard proposed by IBM for specifying (and monitoring) QoS parameters in Web Services. Most SLAs commonly tend to have QoS parameters which are mild variations of the the following measures: response time (latency); availability; maximum allowed query rate (throughput); and security.

To the best of our knowledge, all composition studies consider performance related QoS parameters of contracts in the form of *hard bounds*; two noticeable exceptions are [22] and [12]. For instance, response times and query throughput are required to be less than a certain fixed value and validity of answers to queries must be guaranteed at all times. When composing contracts, hard composition rules are used such as addition or maximum (for response times), or conjunction (for validity of answers to queries).

Whereas this results in elegant and simple composition rules, this general approach by using hard bounds does not fit the reality well. Figure 1 displays

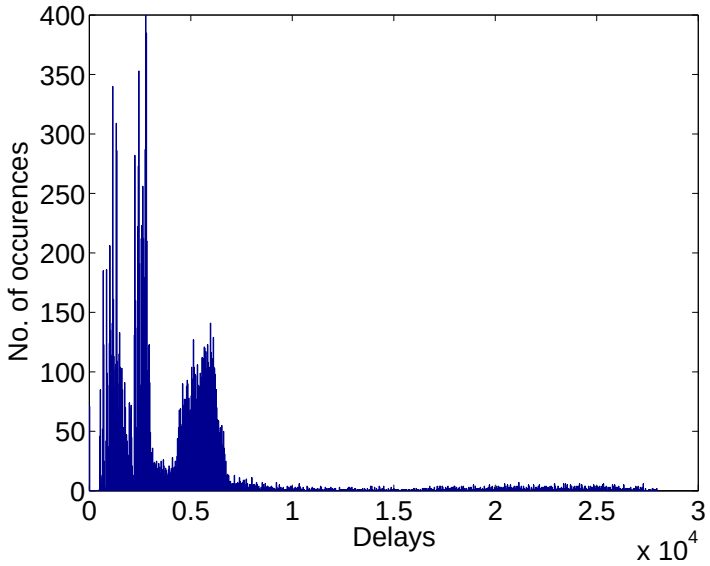

Fig. 1. Measurement records for response times, for Web service `StockQuote`.

a histogram of measured response times for a "StockQuote" Web Service which returns stock prices of a queried entity [30]. These measurements show evidence that the tail of the above distribution cannot be neglected. For example, in this histogram, quantiles of 90%, 95%, and 98%, correspond to response times of 6,494 ms, 13,794 ms, and 23,506 ms respectively. Setting hard bounds in terms of response time would amount to selecting, e.g., the 98% quantile of 23,506 ms, leading to an over pessimistic promise, for this service.

In fact, users would find it very natural to "soften" contracts: a contract should promise, e.g., a response time in less than T milli-sec for 95% of the cases, validity in 99% of the cases, accept a throughput not larger than N queries per second for 98% of a time period of M hours, etc. This sounds reasonable but is not used in practice, partly because soft contracts based on quantiles as above are not supported by composition rules.

In [1] we proposed a probabilistic approach to soft QoS contract composition. Our soft contracts are based on probability distributions. While probabilistic contracts seem, at a first glance, technically involved, we showed that they compose easily. We also showed that the QoS contract for the orchestration obtained by our method are much less pessimistic than those based on hard bound reasoning, thus evidencing opportunities for well sound overbooking. In [12], the authors present a tool for statistical exploration of QoS in composed services in order to detect performance bottlenecks, *i.e.,* components whose replacement would improve overall performance, from an offline point of view. Contract design and algorithms for on-the-fly monitoring are not considered. Moreover, the approach deals only with "box-and-lines" descriptions of workflows; here, we incorporate general orchestrations formally described by programs in the Orc language [24].

The paper is organized as follows. Section 2 briefly reviews probabilistic soft contracts [1]. Then, the problem of soft contract monitoring is set in Section 3. Section 4 is devoted to the presentation of two different statistical monitoring techniques of individual services.

2 Probabilistic soft contracts

For transaction based Web services orchestrations, the situation is the following:

- The orchestration has direct knowledge of the resources of its own server architecture. It knows the traffic it can support, and it can measure the ongoing traffic at a given time.
- The resources and extra traffic for each called Web service is not known to the orchestration—other users of these services belong to the "open world" and the orchestration just ignores their existence.
- The resources and extra traffic for the transport network infrastructure are not known to the orchestration—other traffic belongs to the "open world" and the orchestration just ignores it.

Contracts have therefore emerged as the adequate paradigm for QoS of orchestrations and, more generally, of composite Web services in open world contexts.

A contract consists of an agreement on QoS parameters of the kind listed before. Contracts provide the orchestration with the information needed to construct its own offer to customers. Classically, contracts are formulated as hard bounds on some QoS parameters. As argued in the introduction, it is preferable to characterize contracts in terms of probability distributions over QoS parameters. Hard bounds on parameters will then be replaced by "soft" bounds, of probabilistic or statistical nature. A technique for *probabilistic contract composition* has been developed in [1]. It consists of a Monte-Carlo technique to be applied at design time, for composing contracts with sub-contracting Web services to derive the contract that the orchestration can offer to its customers. This technique is implemented in the TOrQuE (**T**ool for **Or**chestration simulation and **Qu**ality of service **E**valuation) tool , see [1], which performs the following:

1. Starting from an orchestration specified in Orc [24], TOrQuE produces partially ordered executions for it, and labels the corresponding events with QoS attributes.
2. For each called service, samples of answers to queries equipped with their QoS parameters are drawn at random, either from a probability distribution promised by contract, or by re-sampling collected measurements.
3. Each sample of answer to query, randomly produced for the different services, is then fed to the above QoS-enhanced orchestration model, and executed to produce one sample of {orchestration outputs, associated QoS parameters}.
4. These Monte-Carlo runs produce an empirical probability distribution for the tuples consisting of {orchestration outputs, associated QoS parameters}. The latter can be used to establish contracts with customers of the orchestration.

Compared with either techniques using hard bounds, or heuristics manipulating soft contracts, our probabilistic technique was shown to be much less conservative, opening room for well sound overbooking.

3 Monitoring probabilistic soft contracts: problem setting

The very principle of the paradigm of contracts can be stated as "trust and monitor". That is, when setting its own contracts with customers by contract composition following the principles of Section 2, the orchestration will trust the called services that they will meet their contracts. However, life is not so simple and in reality the orchestration must monitor the called services for possible violation of their contracts. Follow up actions can consist of setting financial penalties and/or reconfiguring the orchestration by calling alternative, functionally equivalent, services.

Reconfiguration has been studied by several authors [4,5,9,11,14] in the context of hard contracts where detection is trivial (the considered QoS parameter exceeds the threshold value agreed in the contract). As we said, hard contract monitoring naturally results in drastic policies, possibly leading to excessive decisions—a single contract violation would be seen, by the orchestration, as a fault of the called service. In practice, heuristics would be used to soften such

drastic decisions. Our approach of soft probabilistic contracts offers a mathematically sound framework for soft monitoring of the called services, by the orchestration. In this paper, we illustrate it for the case of response time.

The idea is as follows: Suppose that a given service S has promised, to the orchestration, a probability distribution $\mathbb{P}_S$ for its response time δ. Equivalently, probability distribution $\mathbb{P}_S$ is also characterized by the collection of all its quantiles $F_S(x) =_{\text{def}} \mathbb{P}_S(\delta \leq x)$, for x ranging over the interval of all possible values for the response time. Let the orchestration observe the sequence $\delta_1, \delta_2, \ldots, \delta_n, \ldots$ of actual response times for the service S. Then

$$\widehat{F}_{S,n}(x) =_{\text{def}} \frac{\text{Card}(\{\, \delta_m \mid m \leq n \text{ and } \delta_m \leq x\})}{n} \tag{1}$$

is the proportion of observed response times less than x among the n first queries of service S, by the orchestration (symbol "Card" denotes cardinality of the referred set). Then, informally, the contract is met if, "for n large enough":

$$\forall x \ : \ \widehat{F}_{S,n}(x) \geq F_S(x) \tag{2}$$

holds, *i.e.,* the observed empirical probability that the response time is less than x is *not smaller* than promised.

Statement (2) is too informal, however, since the empirical cumulative distribution function $\widehat{F}_{S,n}(x)$ fluctuates with n, depending on the recorded observations. As a result, rejection decisions would heavily depend on particular abnormal value observed for the response time, something that soft monitoring should precisely avoid. To develop suitable algorithms, we cast our problem in the framework of (sequential) statistical decision theory.

Statistical testing for stochastic dominance [2] provides the adequate mathematical framework to properly state and solve the monitoring problem. Here the problem is stated as follows: let F_S be the cumulative distribution function promised by the service's contract; let F be the actual cumulative distribution function of the service. We wish to decide between the two hypotheses:

$$\begin{gathered} H_0(\text{the contract is met}) : \forall x, F(x) \geq F_S(x) \\ \text{against} \\ H_1(\text{the contract is violated}) : \exists x, F(x) < F_S(x) \end{gathered} \tag{3}$$

In the next section, we propose two different methods for solving statistical decision problem (3). The first method is a *sequential* method. Rather than detecting *if* the contract is violated, the method detects *when* it gets violated.

4 Two statistical methods for soft contract monitoring

4.1 First method: Page-Hinkley Cumulative Sum

Let $F(x) = \mathbb{P}(\delta \leq x)$ be the cumulative distribution function for the response time δ of the considered service. Fix an integer $K > 0$, select K successive quantiles

$$0 < d_1 < \cdots < d_K = +\infty \tag{4}$$

and set, for $k = 1, \ldots, K$:

$$p_k = F(d_k) - F(d_{k-1}) \tag{5}$$

Consider the following statistics:

$$\xi_{k,n} =_{\text{def}} \frac{\text{Card}(\{\, \delta_m \mid m \leq n \text{ and } d_{k-1} < \delta_m \leq d_k \,\})}{n} \tag{6}$$

which counts the proportion of observations sitting between d_{k-1} and d_k, for the n first observations. Let ξ_n be the column vector obtained by stacking the $\xi_{k,n}$, for $k = 1, \ldots, K$. The multivariate Central Limit Theorem [13] states that [4], for

$$\mu^T = \left[\, p_1 \; p_2 \cdots \; p_K \,\right], \tag{7}$$

and n large enough,

$$\zeta_n =_{\text{def}} \sqrt{n}(\xi_n - \mu) \tag{8}$$

is asymptotically distributed as the zero mean Gaussian distribution $\mathcal{N}(0, \Sigma)$, written

$$\zeta_n \sim \mathcal{N}(0, \Sigma), \tag{9}$$

where covariance matrix Σ is equal to

$$\Sigma = \begin{bmatrix} p_1(1-p_1) & -p_1p_2 & \ldots & -p_1p_K \\ -p_2p_1 & p_2(1-p_2) & \ldots & -p_2p_K \\ . & . & \ldots & . \\ . & . & \ldots & . \\ -p_Kp_1 & -p_Kp_2 & \ldots & p_K(1-p_K) \end{bmatrix} \tag{10}$$

The interest of statistics ζ_n is that it (asymptotically) has a known probability distribution, meaning that statistically sound decision procedures can be developed on top of it.

Following (3), let $F_S(x)$ be the cumulative distribution function for the response time of service S. Cumulative distribution function $F_S(x)$ is called the *nominal model* for the response time, and it is assumed known (in our case, it is set by contract). Then, let $F(x)$ be the actual cumulative distribution function of service S. The latter may or may not coincide with $F_S(x)$ but is in any case unknown a priori (we do not know how the called service will actually behave). The orchestration's task is to detect whether or not the contract is met by the called service, *i.e.,* whether or not

$$\text{“}F(x) \text{ sits on the right, not wrong, side of } F_S(x)\text{”} \tag{11}$$

Fix $0 < d_1 < \cdots < d_K = +\infty$ as in (4). Denote by μ and μ_S, and by Σ and Σ_S, the quantities associated with F and F_S via (5), (7) and (10), respectively.

[4] X^T denotes transpose of matrix X.

Problem (11) is formalized as the following statistical decision problem. Let $\theta \in \mathbf{R}^K$ be a design parameter vector (its selection is discussed later). Upon observing the QoS parameters of the considered service, decide

$$\begin{array}{r} H_0(\theta) : \theta^T(\mu - \mu_S) \geq 0 \\ \text{against } H_1(\theta) : \theta^T(\mu - \mu_S) < 0 \end{array} \tag{12}$$

where superscript ^{T} denotes transpose, so that $\theta^T(\mu - \mu_S)$ is the scalar product between the two K-dimensional vectors θ and $\mu - \mu_S$. Note that $\mu - \mu_S$ captures how the actual behaviour of the considered service deviates from the nominal one. How should we select θ? Let $\theta_1, \ldots, \theta_K$ be the components of vector θ. Pick some $k \in \{1, \ldots, K\}$ and take

$$\theta_1 = \cdots = \theta_k = 1 \ , \ \theta_{k+1} = \cdots = \theta_K = 0 \tag{13}$$

Then,

$$\theta^T(\mu - \mu_S) = \sum_{j=1}^{k}(p_j - p_{S,j}) = \sum_{j=1}^{k} p_j - \sum_{j=1}^{k} p_{S,j} = F(d_k) - F_S(d_k)$$

Thus, considering decision problem (12) with design choice (13) amounts to monitoring the kth quantile, for the considered service. The different quantiles can then be monitored by selecting k in (13) accordingly.

The next step is to translate decision problem (12) into some appropriate statistics based on observations. To this end, consider the scalar quantity:

$$\zeta_n^\theta \ =_{\text{def}} \ \theta^T\zeta_n = \sqrt{n}\theta^T(\xi_n - \mu_S) \tag{14}$$

By (9), we know that $\sqrt{n}\theta^T(\xi_n - \mu) \sim \mathcal{N}(0, \sigma_\theta^2)$, which implies $\zeta_n^\theta \sim \mathcal{N}(\widetilde{\mu}_\theta, \sigma_\theta^2)$, where

$$\widetilde{\mu}_\theta =_{\text{def}} \theta^T\sqrt{n}(\mu - \mu_S) \ , \ \sigma_\theta^2 =_{\text{def}} \theta^T\Sigma\theta$$

Therefore,

$$\begin{array}{ll} H_0 & \text{holds iff } \ \widetilde{\mu}_\theta \geq 0 \\ H_1 & \text{holds iff } \ \widetilde{\mu}_\theta < 0 \end{array}$$

Thus our decision problem boils down to that of detecting when the mean of approximately Gaussian scalar random variable ζ_n^θ crosses the zero axis downward. Now, at this point, the covariance matrix Σ is associated with unknown cumulative distribution function F. We could estimate it, but we prefer to simply replace it by Σ_S, which is associated to F_S and can thus be pre-computed prior to running the monitoring procedure. Consider the statistics

$$X_n^\theta = \tfrac{1}{\sigma_\theta}\left(\left(\textstyle\sum_{k=1}^{K} \theta_k \mathbf{1}_{\{d_{k-1} < \delta_n \leq d_k\}}\right) - \theta^T\mu_S\right)$$

We have $\mathbb{E}(X_n^\theta) = \frac{1}{\sigma_\theta}\theta^T(\mu - \mu_S)$, where $\mathbb{E}$ denotes expectation under actual cumulative distribution function F. Also, for n large, we asymptotically have

$$\frac{1}{\sqrt{n}}\sum_{m=1}^{n} X_m^\theta = \frac{1}{\sigma_\theta}\zeta_n^\theta \ \sim \ \mathcal{N}\left(\frac{\widetilde{\mu}_\theta}{\sigma_\theta}, 1\right)$$

Following an argument used in Section 5.4 of [27], we can replace decision Problem (12) by the following asymptotically equivalent one:

$$\text{detect when } \mathbb{E}(X_n^\theta) \text{ becomes negative,}$$

where X_n^θ is considered an independent Gaussian sequence. Thus our decision procedure needs to decide between the two regions $\mathbb{E}(X_n^\theta) \geq 0$ and $\mathbb{E}(X_n^\theta) < 0$.

This raises the well known difficulty in statistics that the two regions are not distant from each other, which causes problem for constructing a meaningful statistics to separate them. The known technique to deal with this is to approximate the two regions $\mathbb{E}(X_n^\theta) \geq 0$ and $\mathbb{E}(X_n^\theta) < 0$ by $\mathbb{E}(X_n^\theta) \geq +\varepsilon$ and $\mathbb{E}(X_n^\theta) \leq -\varepsilon$, where $\varepsilon > 0$ is some small parameter. Accordingly, we consider the following on-line change detection problem for the mean of X_n^θ:

$$\text{detect when } \mathbb{E}(X_n^\theta) \text{ switches from } +\varepsilon \text{ to } -\varepsilon, \tag{15}$$

In (15), ε is a design parameter the tuning of which we discuss below. On-line detection problem (15) is solved by running the following Page-Hinkley cumulative sum test [28,27]:

$$C_0^\theta = 0, \text{ and } C_n^\theta = (C_{n-1}^\theta - X_n^\theta - \varepsilon)_+ \tag{16}$$

where $x_+ =_{\text{def}} \max(x, 0)$. Parameter ε is the tolerance bound, *i.e.,* it states how far below zero $\widetilde{\mu}_\theta$ can be tolerated before detecting contract violation. Now, as long as $\mathbb{E}(X_n^\theta) \geq +\varepsilon$ holds, then C_n^θ sticks to zero with random fluctuations above zero that have a "normalized" size, thanks to the scaling factor used in the definition of X_n^θ. In turn, as soon as $\mathbb{E}(X_n^\theta) \leq -\varepsilon$ occurs (the contract gets violated for the considered quantile), then C_n^θ rapidly grows away from zero and detection occurs. A typical behaviour of C_n^θ is illustrated on Figure 2. Note that,

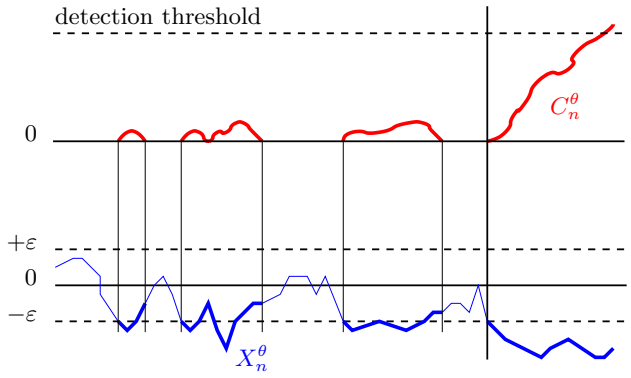

Fig. 2. Joint behaviour of X_n^θ and C_n^θ. The former is shown on the bottom diagram, in blue, as a piecewise linear curve interpolating the successive values $X_{n-1}^\theta, X_n^\theta, X_{n+1}^\theta$, etc. We also show in dashed the two levels $\pm\varepsilon$. Then, we show on top and in red the corresponding behavior of C_n^θ. Note the detection delay.

if θ is selected as in (13), then

$$\mathbb{E}(X_n^\theta) = +\varepsilon \Leftrightarrow F(d_k) - F_0(d_k) = \varepsilon \times \sigma_\theta \tag{17}$$

To monitor a contract consisting of the promised cumulative distribution function F_S, apply (6–10) with the following design choices:

$$K: \text{ user defined; } \; F_0 = F_S; \;\; p_1 = \cdots = p_K = 1/K.$$

Choosing K too small amounts to considering coarsely spaced quantiles, which reduces the accuracy of the monitoring procedure. On the other hand, choosing K too large would result in finely spaced quantiles, which calls for larger n to have enough data in each bin, which in turn result in larger delays for detecting violations. Properly tuning K according the best tradeoff from the designer's viewpoint, is therefore important.

Now, at this point it should be clear that our contract monitoring procedure only uses the K selected quantiles, not the entire cumulative distribution F_S. This also means that the contract itself can be stated in terms of these quantiles, there is no need for agreeing on a fully detailed probability distribution. *Comparing with the common sense single-quantile based contracts, we only need to move to* multi-quantile *contracts, and our comprehensive soft contract approach can be applied.*

4.2 Second method: bootstrapping

Our second method consists in checking in a more direct way that that condition (2) gets violated:

$$\exists x \;:\; \widehat{F}_{S,n}(x) < F_S(x) \tag{18}$$

The problem with detection rule (18) are the random fluctuations of $\widehat{F}_{S,n}(x)$. Thus we introduce some tolerance zone in (18), by strengthening it as $\exists x : \widehat{F}_{S,n}(x) < F_S(x) - \lambda$, or, equivalently

$$\sup_x (F_S(x) - \widehat{F}_{S,n}(x)) \geq \lambda \tag{19}$$

where λ is a small positive parameter. Thus the remaining issue is to tune λ in a statistically meaningful way. This is achieved by applying the following bootstrapping procedure:

1. As sketched in Section 2 (see [1] for details), at design time, the contract of the orchestration is evaluated by performing Monte-Carlo based contract composition. To this end, sample response times are drawn for Monte-Carlo simulations, based on the assumed cumulative distribution function $F_S(x)$ for the service. To prepare for monitoring, in addition, we produce bootstrap estimates of the cumulative distribution function $\widehat{F}_{S,\gamma}(x)$ for this service:

$$\widehat{F}_{S,\gamma}(x) = \frac{\text{Card}(\{\, \delta_m \mid m \in \gamma \text{ and } \delta_m \leq x\})}{|\gamma|},$$

where γ is a randomly selected subset of $\{1, \ldots, n\}$, the index set of the random data generated, based on the assumed cumulative distribution function $F_S(x)$ for the service. Let Γ be the set of such γ's.

2. At design time, select a level (e.g., 95%). Regard $\widehat{F}_{S,\gamma}(x)$ as a population, where γ ranges over Γ. Select threshold λ such that the following holds, for 95% of the $\gamma \in \Gamma$:
$$\sup_x (F_S(x) - \widehat{F}_{S,\gamma}(x)) \leq \lambda$$

3. At run time, compute $\widehat{F}_{S,n}(x)$ as in formula (1) and decide upon violation if (19) occurs.

Unlike our method of Section 4.1, decision rule (19) is not sequential. To get a sequential and on-line monitoring procedure, a possibility is to run (19) for $\widehat{F}_{S,n,m}(x)$ computed over successive overlapping windows of data: $\{1, \ldots, n\}$, $\{p, \ldots, p+n\}$, $\ldots$, $\{mp, \ldots, mp+n\}$, and so on, for n fixed, $p \leq n$ fixed (to ensure that windows overlap), and $m = 1, 2, \ldots$

5 Discussion and Future Work

We have presented a QoS contract monitoring technique, based on the new concept of soft probabilistic contract. Contract monitoring must be soft too (we do not want to be harsh in rejecting a service because of an isolated, outlier QoS parameter value, but we want that repeated defaults lead to rejection). Probabilistic contracts provide the rationale for developing soft monitoring in this sense.

The two techniques we propose are one-sided statistical tests. The first one is a parametric test. It consists in monitoring a finite pre-specified set of quantiles. It takes advantage of the Central Limit Theorem to ensure proper calibration of the test statistics, which in turn avoids tedious tuning of highly non robust parameters for the decision procedure. The second one is non parametric and relies on bootstrapping for tuning the associated rejection region. These two techniques are currently being evaluated on Monte-Carlo simulations as well as real data from actual Web services.

Two problems are left aside in the present study. First, the transport network layer is not considered, which biases QoS performance of the called services—the transport network itself participates to the QoS perceived by the orchestration. Second, and more importantly, our present technique treats all called services on an equal basis. While this may seem fair at a first glance, it is not rational from the point of view of the orchestration. A service whose QoS has little influence on the overall QoS of the orchestration should be treated differently from a service whose QoS is critical for the orchestration to meet its own QoS promise. A method to compensate for this by taking advantage of the QoS-enhanced model of the orchestration is under development and will be reported in future work.

References

1. S. Rosario, A. Benveniste, S. Haar, and C. Jard. Probabilistic QoS and soft contracts for transaction based Web services orchestrations. Submitted for publication. Feb. 2007. http://www.irisa.fr/distribcom/benveniste/pub/QoSWS2007.html
2. G. Anderson. Nonparametric tests of stochastic dominance in income distributions. *Econometrica,* **64**(5), 1183–1193, 1996.
3. W.M.P. van der Aalst and K.M. van Hee. Workflow Management: Models, Methods, and Systems. MIT Press, Cambridge, MA, USA, 2002.
4. F. Barbon, P. Traverso, M. Pistore, and M. Trainotti. Run-Time Monitoring of Instances and Classes of Web Service Compositions. In *Proc. of Int. Conf. on Web Services*, ICWS06, F. Leymann and L.J. Zhang, Eds. September 18-22, 2006, Chicago, USA. 2006.
5. S. Bhiri, W. Gaaloul, and C. Godart. Discovering and Improving Recovery Mechanisms of Composite Web Services In *Proc. of Int. Conf. on Web Services*, ICWS06, F. Leymann and L.J. Zhang, Eds. September 18-22, 2006, Chicago, USA. 2006.
6. P. Bhoj, S. Singhal, and S. Chutani. SLA Management in Federated Environments. In M. Sloman, S. Mazumdar, and E. Lupu, editors, *Proc. of Sixth IFIP/IEEE Symposium on Integrated Network Management,* IM 99, pages 293-308.
7. T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte (Editor), I. Trickovic, and S. Weerawarana. Business Process Execution Language for Web Services. [BPEL4WS.] Version 1.1. 5-May-2003. 136 pages. http://xml.coverpages.org/BPELv11-May052003Final.pdf
8. W.M.P van der Aalst. Verification of workflow nets. *Application and Theory of Petri Nets 1997*, volume 1248 of LNCS, pages 407426.
9. G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani. QoS-Aware Replanning of Composite Web Services. In *Proc. of Int. Conf. on Web Services*, ICWS05, C.K. Chang, L-J. Zhang, Eds. July 11-15, 2005, Orlando, USA.
10. Andries van Dijk. Contracting Workflows and Protocol Patterns. Business Process Management 2003, Wil M. P. van der Aalst and Arthur H. M. ter Hofstede and Mathias Weske Eds., LNCS 2678, 152-167, 2003.
11. A. Erradi, P. Maheshwari, and V. Tosic. Recovery Policies for Enhancing Web Services Reliability. In *Proc. of Int. Conf. on Web Services*, ICWS06, F. Leymann and L.J. Zhang, Eds. September 18-22, 2006, Chicago, USA.
12. C. Hughes and J. Hillman. QoS Explorer: A tool for exploring QoS in Composed Services. Proc. ICWS 2006, 797–804.
13. M.G. Kendall and A. Stewart. *The Advanced Theory of Statistics.* London, Griffen. 1979.
14. Y. Li, K. Sun, J. Qiu, and Y. Chen. Self-Reconfiguration of Service-Based Systems: A Case Study for Service Level Agreements and Resource Optimization. In *Proc. of Int. Conf. on Web Services*, ICWS05, C.K. Chang, L-J. Zhang, Eds. July 11-15, 2005, Orlando, USA.
15. F. Puhlmann, M. Weske. Using the Pi-Calculus for Formalizing Workflow Patterns. In W.M.P. van der Aalst et al. (Eds.): BPM 2005, volume 3649 of LNCS, Nancy, France, Springer-Verlag (2005) 153-168.
16. C. Ouyang, E. Verbeek, W.M.P van der Aalst and S. Breutel. Formal Semantics and Analysis of Control Flow in WS-BPEL. BPM Center Report BPM-05-15, BPMcenter.org, 2005.
17. W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow Mining: A Survey of Issues and Approaches. *Data and Knowledge Engineering,* 47(2):237-267, 2003.

18. J. Arias-Fisteus, L. Sánchez Fernández, and C. Delgado Kloos. Applying model checking to BPEL4WS business collaborations. SAC 2005: 826-830.
19. Keller A., Ludwig H., The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. *Journal of Network and Systems Management*, Vol. 11, No 1, Plenum Publishing, pp. 57-81.
20. N. Kavantzas, D. Burdett, G. Ritzinger, T. Fletcher, Y. Lafon. Web Services Choreography Description Language – WS-CDL, version 1.0. http://www.w3.org/2002/ws/chor/edcopies/cdl/cdl.html
21. J. Cardoso, A. Sheth, J. Miller, J. Arnold, and K. Kochut, Quality of Service for Workflows and Web Service Processes, Journal of Web Semantics 2004.
22. Z. Liu, M. S. Squillante, and J. L. Wolf. On Maximizing Service-Level-Agreement Profits. *ACM E-Commerce Conference*, Tampa, FL. October 2001
23. A.C. Davison, and D.V. Hinkley. Bootstrap Methods and their Application. Cambridge University Press, 1997.
24. Jayadev Misra and William Cook. Computation orchestration: A Basis for Wide-Area Computing. *Journal of Software and Systems Modeling,* May 2006. http://www.cs.utexas.edu/~wcook/papers/OrcJSSM05/OrcJSSM.pdf
25. S. Rosario. D. Kitchin, A. Benveniste, W. Cook, S. Haar, and C. Jard. Event Structure Semantics for Orc. submitted, 2007.
26. A. Sahai, V. Machiraju, M. Sayal, Aad P. A. van Moorsel, and F. Casati. Automated SLA Monitoring for Web Services. DSOM 2002: 28-41.
27. A. Benveniste, M. Métivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations.* Applications of Mathematics, 22. Springer Verlag. 1990.
28. M. Basseville and I.V. Nikiforov. *Detection of Abrupt Changes, Theory and Applications.* Prentice Hall, 1993. Also available from http://www.irisa.fr/sisthem/kniga
29. H. G. Song and K. Lee: sPAC (Web Services Performance Analysis Center): Performance Analysis and Estimation Tool of Web Services. Business Process Management 2005: 109-119.
30. XMethods. http://www.xmethods.net