knowledgeweb
realizing the semantic web

D2.4.11 Reputation-based Service Level Agreements and Decentralized Orchestration of Composite Services

Coordinator: Radu Jurca (EPFL)
Boi Faltings (EPFL),
Walter Binder (EPFL),
David Portabella Clotet (EPFL)

Abstract.
EU-IST Network of Excellence (NoE) IST-2004-507482 KWEB
Deliverable D2.4.11 v1 (WP2.4)

Service-oriented computing enables the construction of distributed applications by integrating services that are available over the web. In this deliverable we describe a simple, robust mechanism that eliminates incentives for selfish providers and clients in a service-oriented environment to cheat. In the second version of this deliverable we will describe also schemes for executing composite web services communicating in an efficient decentralized way.
Keyword list: Reputation Mechanisms, Sevice Level Agreements, Decentralized Orchestration, Web Services

Document Identifier	KWEB/2006/D2.4.11/v1
Project	KWEB EU-IST-2004-507482
Version	v1
Date	December 21, 2006
State	Final
Distribution	public

Knowledge Web Consortium

This document is part of a research project funded by the IST Programme of the Commission of the European Communities as project number IST-2004-507482.

University of Innsbruck (UIBK) - Coordinator
Institute of Computer Science
Technikerstrasse 13
A-6020 Innsbruck
Austria
Contact person: Dieter Fensel
E-mail address: dieter.fensel@uibk.ac.at

École Polytechnique Fédérale de Lausanne (EPFL)
Computer Science Department
Swiss Federal Institute of Technology
IN (Ecublens), CH-1015 Lausanne
Switzerland
Contact person: Boi Faltings
E-mail address: boi.faltings@epfl.ch

France Telecom (FT)
4 Rue du Clos Courtel
35512 Cesson Sévigné
France. PO Box 91226
Contact person : Alain Leger
E-mail address: alain.leger@rd.francetelecom.com

Freie Universität Berlin (FU Berlin)
Takustrasse 9
14195 Berlin
Germany
Contact person: Robert Tolksdorf
E-mail address: tolk@inf.fu-berlin.de

Free University of Bozen-Bolzano (FUB)
Piazza Domenicani 3
39100 Bolzano
Italy
Contact person: Enrico Franconi
E-mail address: franconi@inf.unibz.it

Institut National de Recherche en Informatique et en Automatique (INRIA)
ZIRST - 655 avenue de l'Europe -
Montbonnot Saint Martin
38334 Saint-Ismier
France
Contact person: Jérôme Euzenat
E-mail address: Jerome.Euzenat@inrialpes.fr

Centre for Research and Technology Hellas / Informatics and Telematics Institute (ITI-CERTH)
1st km Thermi - Panorama road
57001 Thermi-Thessaloniki
Greece. Po Box 361
Contact person: Michael G. Strintzis
E-mail address: strintzi@iti.gr

Learning Lab Lower Saxony (L3S)
Expo Plaza 1
30539 Hannover
Germany
Contact person: Wolfgang Nejdl
E-mail address: nejdl@learninglab.de

National University of Ireland Galway (NUIG)
National University of Ireland
Science and Technology Building
University Road
Galway
Ireland
Contact person: Christoph Bussler
E-mail address: chris.bussler@deri.ie

The Open University (OU)
Knowledge Media Institute
The Open University
Milton Keynes, MK7 6AA
United Kingdom
Contact person: Enrico Motta
E-mail address: e.motta@open.ac.uk

Universidad Politécnica de Madrid (UPM)
Campus de Montegancedo sn
28660 Boadilla del Monte
Spain
Contact person: Asunción Gómez Pérez
E-mail address: asun@fi.upm.es

University of Karlsruhe (UKARL)
Institut für Angewandte Informatik und Formale
Beschreibungsverfahren - AIFB
Universität Karlsruhe
D-76128 Karlsruhe
Germany
Contact person: Rudi Studer
E-mail address: studer@aifb.uni-karlsruhe.de

University of Liverpool (UniLiv)
Chadwick Building, Peach Street
L697ZF Liverpool
United Kingdom
Contact person: Michael Wooldridge
E-mail address: M.J.Wooldridge@csc.liv.ac.uk

University of Manchester (UoM)
Room 2.32. Kilburn Building, Department of Computer
Science, University of Manchester, Oxford Road
Manchester, M13 9PL
United Kingdom
Contact person: Carole Goble
E-mail address: carole@cs.man.ac.uk

University of Sheffield (USFD)
Regent Court, 211 Portobello street
S14DP Sheffield
United Kingdom
Contact person: Hamish Cunningham
E-mail address: hamish@dcs.shef.ac.uk

University of Trento (UniTn)
Via Sommarive 14
38050 Trento
Italy
Contact person: Fausto Giunchiglia
E-mail address: fausto@dit.unitn.it

Vrije Universiteit Amsterdam (VUA)
De Boelelaan 1081a
1081HV. Amsterdam
The Netherlands
Contact person: Frank van Harmelen
E-mail address: Frank.van.Harmelen@cs.vu.nl

Vrije Universiteit Brussel (VUB)
Pleinlaan 2, Building G10
1050 Brussels
Belgium
Contact person: Robert Meersman
E-mail address: robert.meersman@vub.ac.be

Work package participants

The following partners have taken an active part in the work leading to the elaboration of this document:

Ecole Polytechnique Fédérale de Lausanne

Changes

Version	Date	Author	Changes
0.1	02.06.06	David Portabella Clotet	creation
0.3	29.06.06	Radu Jurca	Sections: Reputation-based SLA
0.4	07.07.06	Radu Jurca	Sections: The Setting and Interaction Protocol
0.6	15.08.06	Radu Jurca	Sections: Truthfull reporting and enforcing
0.6	19.09.06	Radu Jurca	Section: Collusion
0.7	20.09.06	Walter Binder	Introduction: Chapter Decentralized Orchestration
0.8	07.11.06	Radu Jurca	Section: Experimental evaluation
1.0	18.12.06	Radu Jurca	Introduction, Conclusion & Related Work

Executive Summary

Service-oriented computing enables the construction of distributed applications by integrating services that are available over the web [15]. However, in such environments, malicious providers may advertise false service capabilities. It is therefore important to study mechanisms to avoid this type of behaviour. A second problem adressed in this deliverable is the inefficient routing of messages in centralized orchestration of composite web services.

We describe a simple, robust mechanism that eliminates incentives for selfish providers and clients to cheat, while the assumptions behind the mechanism are fairly general, making it a candidate for many practical settings.

In the second version of this deliverable we will describe also schemes for executing composite web services communicating in an efficient decentralized way.

Contents

Chapter 1

Introduction

Service oriented computing systems represent an attractive paradigm for the business world of tomorrow. User requests ranging from trip reservations to complex optimization problems, are no longer atomically treated by monolithic organizations, but rather decomposed into smaller components that are separately addressed by different service providers [18]. While the advantages of such a scenario are clear (simplicity, ease of management and customization, fault tolerance and scalability), the fact that services are delivered by independent, self-interested providers poses new challenges, such as trustworthiness and decentralized orchestration of composite web services.

Most web services need to be contracted through service level agreements that typically specify a certain quality of service (QoS) in return for a certain price. We propose a new form of service level agreement where the price is determined by the QoS actually delivered. We show that such agreements make it optimal for the service provider to deliver the service at the promised quality. To allow efficient monitoring of the actual QoS, we introduce a reputation mechanism. A scoring rule makes it optimal for the users of a service to correctly report the QoS they observed. Thus, we obtain a practical scheme for service-level agreements that makes it uninteresting for providers to deviate from their best effort.

Traditional, centralized orchestration of composite web services often leads to inefficient routing of messages. To solve this problem, we present a novel scheme to execute composite web services in a fully decentralized way. We introduce service invocation triggers, a lightweight infrastructure that routes messages directly from the producing service to the consuming one, enabling fully decentralized orchestration. We will measure total network traffic, comparing centralized workflow orchestration versus decentralized orch.

In Chapter 2 we describe a practical scheme for service level agreements based on reputation mechanisms. In Chapter 3 we will describe a scheme to execute compostive services in a decentralized way. We conclude with Chapter 4.

Chapter 2

Reputation-based Service Level Agreements for Web Services

2.1 Introduction

Service oriented computing systems represent an attractive paradigm for the business world of tomorrow. User requests ranging from trip reservations to complex optimization problems, are no longer atomically treated by monolithic organizations, but rather decomposed into smaller components that are separately addressed by different service providers [18]. While the advantages of such a scenario are clear (simplicity, ease of management and customization, fault tolerance and scalability), the fact that services are delivered by independent, self-interested providers poses new challenges.

We assume a scenario where services are contracted through Service Level Agreements (SLAs) that specify a certain quality of service (QoS) in return for a certain price. Independent monitoring of QoS is expensive and technically difficult. Without proper monitoring, selfish service providers can increase their revenues by cheating: they advertise high quality but do not invest the necessary effort to provision the service. Anticipating this behavior, rational clients will not trust the providers, and therefore, will decrease to a minimum the amounts they are willing to pay for the service. Such a market is very inefficient, and will drive away trustworthy providers.

In this chapter, we consider scenarios where a group of customers are treated identically by the provider using the same service level agreement. In this case, the SLA can be based on the service provided to them as a group. Given correct information about the QoS, such agreements make it optimal for the service provider to deliver at least the advertised quality to each participant.

This leaves the problem of monitoring this quality of service. As a second main result, we show that independent monitoring can actually be replaced by a reputation system where monitoring is done by the customers themselves. This raises the problems of (a)

eliciting honest feedback from clients and (b) preventing collusion. We show how a reputation mechanism can use side-payments (i.e. clients get paid for submitting feedback) to make it rational for all clients to truthfully share their feedback. Moreover, when a reputation mechanism has a small number of "trusted" reports (i.e. feedback that is true with high probability) we prove that rational clients will not collude in order to artificially decrease the reputation of a service provider.

This chapter thus describes a practical mechanism that eliminates incentives for selfish service providers to cheat while greatly reducing the QoS monitoring burden on the market. The scheme is safe against strategic lying and bad-mouthing[1] collusion. Section 2.2 formally describes the setting and the assumptions behind our results, Section 2.4 describes in detail the service level agreements and their properties while Section 2.5 addresses the problem of truthful reporting. Section 2.8 evaluates our mechanism, followed by related work and a conclusion.

2.2 The Setting

We consider an online market where service providers repeatedly offer the same service to the interested clients, in exchange for money. The transactions between service providers and clients are regulated by a Service Level Agreement (SLA) that defines (among others) quality parameters of the delivered service (i.e. the QoS) and the dependence of price on the actual QoS. When there are several QoS parameters, we assume that the SLA can be split into separate agreements for each parameter such that the price is the sum of the prices in the individual SLAs. A precise definition of the SLA for our mechanism is given in Section 2.4, Definition 2.4. A practical framework supporting such interactions is described in detail by Dan et al. [3].

We assume there is a large enough group of clients that share the same QoS and SLA during a predefined period of time. Note that a provider can have several customer groups (e.g. silver/gold/platinium customers), as far as all clients in a certain group are treated identically. Therefore, the average satisfaction rate of the customers in a given group, in a given period of time, can be used to estimate the real QoS delivered by the provider. We denote by $\mathcal{Q}$ the set of all possible values for the QoS.

We assume that clients have two degrees of satisfaction: they either perceive *high* quality or *low* quality service. High quality service, for example, is perceived when the answer to the service request is received before a specified deadline. This binary model can be easily extended to finer grained quality levels and multiple quality parameters.

The market has an independent *reputation mechanism* (RM) that collects binary feedback from clients. "1" denotes positive feedback and signals the fact that the client has observed a high quality service. Likewise, "0" denotes negative feedback and signals low quality service. Feedback is collected at the end of each time period, when all transactions

[1] strategic denigration of a provider's reputation through false negative feedback

are assumed completed. The reputation of a provider is computed by the RM as the percentage of positive reports submitted by the members of a particular customer group, in a given period. Reputation, therefore, equals the average QoS delivered to a given customer group in a given period.

Clients can make involuntary mistakes when submitting feedback. When q percent of the clients perceive high quality, the reputation of the provider equals $q + \eta_r$; the noise η_r is assumed normally distributed around 0 with variance σ_r^2.

We further assume that the RM can (a) pay clients for submitting reports, and (b) obtain a limited number of trusted reports that are true with high probability. Trusted reports can be obtained from specialized agents[2] hired to anonymously test the service delivered by the provider. In Section 2.5 we show how side payments and trusted reports can be used to elicit honest feedback from rational clients, and prevent collusion.

Service providers differ in their ability and knowledge to provide qualitative services. For example, the time required to successfully answer a service invocation (up to some random noise) depends on the available infrastructure (e.g. hardware, software, network capacity) and on the number of requests accepted by the provider in a given time window.

The infrastructure is assumed fixed and defines the *type* of the provider. Two providers have the same type if they have exactly the same capabilities for providing service. Formally, the set of possible types is denoted by Θ, and members of this set are denoted as θ.

The number of accepted requests, on the other hand, can be strategically decided by the service provider. Given the available infrastructure (i.e. a type), the provider needs to limit the number of accepted requests in order to deliver the required answers before the deadline, with high probability. Providing high QoS requires *effort* (e.g. limiting requests and giving up revenue), and hence, has a cost.

Let $c(\theta, e)$ be the cost incurred by a provider of type θ when exerting effort e in a given period of time. The cost function is private to each provider type, and usually concave (i.e. higher quality demands increasingly more effort). However, our results are independent of the form of the cost function.

The provider's type (e.g. available infrastructure) and effort (e.g. number of accepted requests) determine the actual QoS provided to clients. If we denote by $\mathcal{E}$ the set of possible effort levels, and by $\mathcal{Q}$ the set of possible quality levels, let the function $\phi : \Theta \times \mathcal{E} \rightarrow \mathcal{Q}$ defines the mapping between type, effort and QoS. External factors and noise also influence the QoS. A type θ provider will therefore deliver quality $\phi(\theta, e) + \eta_n$ when exerting effort e. η_n is assumed normally distributed around 0 with variance σ_n^2.

[2] sites like Keynote Systems (www.keynote.com) and Xaffire Inc. (www.xaffire.com) offer such services.

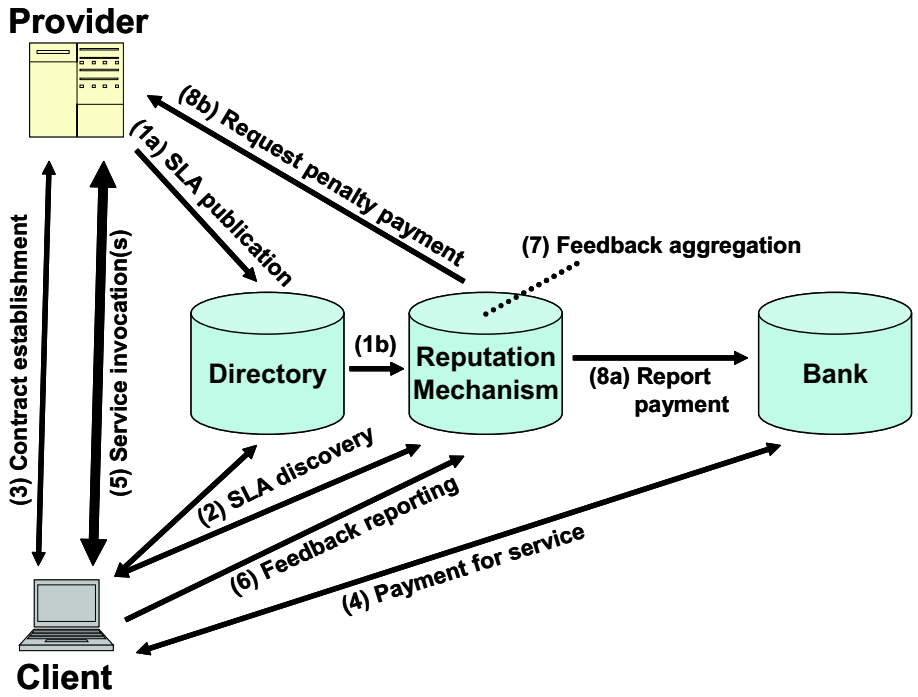

Figure 2.1: Interaction protocol involving a RM.

2.3 Interaction Protocol

The participants in our environment are the following: *service providers* advertise SLAs and offer the corresponding services; *clients* choose SLAs and invoke the respective services; *service directories* facilitate the matching between clients and providers; *RMs* collect and aggregate feedback from the clients; a *bank* handles payments. The RMs and the bank are trusted parties. A RM can be integrated into a service directory in order to enable efficient, reputation-aware SLA selection. In this case, the service directory integrating a RM is assumed to be trusted.

Fig. 2.1 illustrates the interactions between the aforementioned participants:

1. Providers advertise SLAs to a service directory (1a). Each SLA uniquely identifies the service provider and the service functionality, for example by referring to a WSDL service description, and defines the price and QoS for service invocation. The service directory assigns a suitable RM for each SLA advertisement, which shall be used for feedback reporting. The instantiation of a RM for a new SLA (1b) requires solving a linear optimization problem. Advertised SLAs remain valid for a period of time specified by the provider. After expiration, they are removed from the directory. Service directories may support leases, allowing service providers to refresh SLA advertisements. Each SLA receives a unique *SLA-ID*, computed as a secure hashcode of the SLA.

2. Clients search for advertised SLAs according to functional and non-functional criteria, as well as according to reputation information. To this end, clients access a

directory and a RM. If the RM is integrated within the directory, reputation-based filtering constraints can be directly included in the directory query. Clients may inspect reputation information specific to a SLA, or aggregated reputation information for a service provider.

3. The client and the chosen provider establish a contract for a given SLA, for a given period of time. The client sends a request message to the service provider, including *Client-ID*, *SLA-ID*, and the number of requested service invocations, *Nr-Invoc*. The service provider may reject the request, if it (temporarily) cannot meet the conditions of the SLA. The response message sent by the service provider is a non-forgeable service invocation capability (SIC), valid for *Nr-Invoc* service invocations according to the conditions advertised in the SLA *SLA-ID*. The SIC will also be used by the client to report feedback.

4. The client pays for the agreed number of service invocations (i.e., *Nr-Invoc* times the price stated within the SLA). The payment message includes the SIC, and the bank returns the signed SIC in order to certify successful payment.

5. The client requests the service, and the provider responds. For each service invocation, the client has to provide a valid SIC signed by the bank. Hence, the service provider can easily determine that the client has payed for the SLA. The service provider keeps track of the number of service invocations for each valid SIC in order to ensure that this number does not exceed the contracted *Nr-Invoc* value. The client monitors the QoS parameters to be reported to the RM.

6. The client sends feedback to the RM. The feedback contains the SIC signed by the bank, and a timestamped series of quality reports. For each SIC, the client may send between 1 and *Nr-Invoc* reports. The quality reports need not necessarily be aggregated within a single message. I.e., for the same SIC, the client may send several messages with a varying number of quality reports. The RM does not verify whether a service was actually invoked by the client, but it ensures that the client paid for the invocation. I.e., the RM rejects reports if the SIC has not been signed by the bank.

7. The RM aggregates received feedback at the end of each time period. From all valid quality reports about a SLA, the RM estimates the actually delivered QoS by computing the distribution of values (i.e., histogram) for every quality attribute described by the SLA. Feedback can also be used to update the reputation of the service provider.

8. The RM pays valid reports. Finally, the RM publishes the monitored QoS value for the current period and notifies the providers about the penalties they must pay (8b). Service providers who do not pay the agreed penalties may be put on a "black list" by the RM and consequently will be avoided by clients upon service selection.

2.4 Reputation-based Service Level Agreements

The idea behind the SLA we propose in this chapter to make higher, untruthful, advertisements of QoS unprofitable for service providers. For that, our SLA follows the framework

proposed in [3] and specifies a monetary penalty that must be paid by the provider to each client at the end of a given period of time. The penalty is directly proportional to the difference between promised and delivered QoS, such that the total revenue of a provider declaring higher QoS (i.e. the price of the advertised QoS minus the penalty for providing lower QoS) is lower than the price obtained from truthfully declaring the intended QoS in the first place. The novelty of our approach is that we use reputation information to compute the penalties paid by providers.

A reputation-based Service Level Agreement states the following terms:

- `per_validity`: the period of validity. Time is indexed according to a discrete variable t;
- `cust_group`: the intended customer group (e.g. silver/gold/platinium customers);
- `QoS` (denoted as $\bar{q}_t \in \mathcal{Q}$): the quality of service (e.g. the average probability of delivering high quality service);
- `price` (denoted as p_t) : the price of service;
- `penalty`: the reputation-based penalty to be paid by the provider to the client for deviating from the terms of the SLA. The penalty $\lambda_t : \mathcal{Q} \times \mathcal{Q} \rightarrow \mathbb{R}^+$ is a function of advertised QoS (i.e. $\bar{q}_t$) and delivered QoS (i.e. the reputation, R_t). $\lambda_t(\bar{q}_t, R_t) = 0$ for all $R_t \geq \bar{q}_t$ and strictly positive otherwise.

The SLA is defined by the service provider prior to the period of time, t, when the SLA is valid. The provider chooses (a) the advertised QoS (i.e. $\bar{q}_t$), (b) the price charged for service (i.e. p_t), (c) the penalty function (i.e. $\lambda_t(\cdot, \cdot)$), and (d) the exerted effort (i.e. e_t). The first three choices are made public through the SLA (we therefore use the shorthand notation: $sla_t = (\bar{q}_t, p_t, \lambda_t)$) while the forth one is kept private.

As a first result we derive sufficient constraints on the penalty function such that service providers of all types find it optimal to deliver at least the promised QoS. As expected, these constraints are related to the market price of QoS.

Proposition 1 *Let the function $u : \mathcal{Q} \rightarrow \mathbb{R}$ define the market price clients pay for a given QoS. When (1) clients truthfully submit feedback, and (2) the penalty function satisfies: $\partial\lambda(q, R)/\partial q \geq 2u'(q)$, for all q and R, the reputation-based SLA makes it rational for all service provider types to deliver at least the advertised QoS.*

PROOF.

Consider a type θ provider advertising $sla_t = (\bar{q}_t, p_t, \lambda_t)$ in period t. If the provider exerts effort level e_t, his expected revenue is:

$$V_t(e_t, \bar{q}_t) = N_t \cdot \big(p_t - E[\lambda(\bar{q}_t, R_t)]\big) - c(e_t, \theta); \tag{2.1}$$

where R_t is the reputation of the provider at the end of time period t, N_t is the number of services sold in period t, $c(e_t, \theta)$ is the cost of effort, and the expected penalty is computed with respect to possible values of R_t. V_t does not depend on any past or future decisions of the provider. By individually maximizing the sequence of payoffs, a rational provider also maximizes his life-time revenue.

When the provider exerts effort e_t, the quality of the service equals $\phi(\theta, e_t) + \eta_n$, where η_n is normally distributed around 0 with variance σ_n^2. Clients truthfully report their observations, however, they make mistakes. Assuming that the number of reports is big enough, the value of the reputation $R_t = \phi(\theta, e_t) + \eta_n + \eta_r$ is normally distributed around $\phi(\theta, e_t)$ with the variance $\sigma^2 = \sigma_n^2 + \sigma_r^2$.

Let $(e^*, q^*) = \arg\max_{(e_t, \bar{q}_t)} E\big[V_t(e_t, \bar{q}_t)\big]$ be the optimal effort level and advertised QoS. Assuming the provider asks the maximum price for the advertised quality (i.e. $p_t = u(\bar{q}_t)$), the first order condition on q^* becomes:

$$\begin{aligned}\frac{1}{N_t}\frac{\partial V_t}{\partial \bar{q}_t}(e^*, q^*) &= u'(q^*) - E\Big[\frac{\partial \lambda}{\partial \bar{q}_t}(q^*, \phi(e^*) + \eta)\Big] \\ &= u'(q^*) - \int_{q<q^*} normpdf(q|\phi(e^*), \sigma)\frac{\partial \lambda}{\partial \bar{q}_t}(q^*, q)dq = 0;\end{aligned}$$

where $normpdf(q|\phi(e^*), \sigma)$ is the normal probability distribution function with the mean $\phi(e^*)$ and variance σ^2.

By replacing the condition on λ, we get:

$$\int_{q<q^*} normpdf(q|\phi(e^*), \sigma)dq \leq 0.5 \tag{2.2}$$

i.e. the cumulative probability distribution $Pr[q < q^*|\phi(e^*)] \leq 0.5$. For a normal distribution, this is only true if $q^* \leq \phi(e^*)$. In other words, all provider types deliver at least the promised QoS. □

Clients can check the constraint on the penalty function by analyzing the previous transactions concluded in the market. For every previously negotiated $sla_i = (\bar{q}_i, p_i, \lambda_i)$, clients infer that the market price corresponding to $\bar{q}_i$ must be higher than p_i: i.e. $u(\bar{q}_i) \geq p_i$. Previous interactions thus establish a lower bound on the real market price that can be used to safe-check the validity of the penalty function. Please note that the proof above does not make any assumptions about the market price or the cost function of the providers. Reputation-based SLAs can thus be used for a variety of settings.

All service providers have the incentive to minimize the penalty function specified by the SLA. This happens when the constraint in Proposition 1 is satisfied up to equality. As an immediate consequence, all service providers advertise exactly the intended QoS (Equation 2.2).

The mechanism assumes that (1) clients submit honest feedback, (2) they are able to submit feedback only after having interacted with the provider, and (3) they submit only one feedback per transaction. The first assumption can be integrated into the broader context of truthful feedback elicitation. The problem can be solved by side-payments (i.e. clients get paid by the reputation mechanism for submitting feedback) and will be addressed in more details in Section 2.5.

The second and third assumptions can be implemented through cryptographic mechanisms based on a public key infrastructure. As part of the interaction, providers can deliver signed one-time certificates that can later be used by clients to provide feedback. A concrete implementation of such a security mechanism for reputation mechanisms is presented in [7].

2.5 Truthful Reporting

Reporting honest feedback (as required by the proof of Proposition 1) is not exactly in the best interest of rational clients. By reporting false negative feedback (when she actually experienced a successful service) a client decreases the reputation of the provider, and consequently decreases the overall price (i.e. price minus penalty) she needs to pay for the service. Actually, it is always in the clients' best interest to report negative feedback. Unless this strategic bias can be eliminated, rational clients will consistently downrate providers who will eventually quit the market.

Side-payments (i.e. clients get paid for submitting feedback) can be designed to encourage rational clients to report the truth. This is possible because the observation of a client (i.e. the fact that the service delivered to her had high or low quality) slightly changes the client's belief regarding the experience of future clients. Take a client having experienced a low quality service (e.g. a request failure). The client will infer that the present invocation failure is likely to be caused by a problem affecting the general infrastructure of the provider. Future clients will probably be affected by the failure as well, and therefore, the average QoS experienced by the next clients is slightly lower than expected (prior to observing the failure).

Similarly, a high quality service testifies for the well functioning of the provider's infrastructure and encourages more optimistic estimates regarding the QoS observed by future clients. This asymmetry in the beliefs regarding the experience of future clients can be exploited by side-payments that make truthful reporting optimal.

Concretely, we adapt the mechanism described by Miller et al. [13] to our setting. The basic idea behind the mechanism is to use the feedback of a future client (referred to as *rater*) to rate (and compute the payment for) a submitted report. The present report is used to update a probability distribution for the report of the rater. The payment for the report is then computed by comparing the *likelihood* assigned to the rater's rating with the rater's actual rating.

$S(0,0)$	$\frac{2(1-\bar{q}_t)(1-2\bar{q}_t+\bar{q}_t^2+\sigma^2)-(\bar{q}_t-\bar{q}_t^2-\sigma^2)^2-(1-2\bar{q}_t+\bar{q}_t^2+\sigma^2)^2}{(1-\bar{q}_t)^2}$
$S(1,0)$	$\frac{2(1-\bar{q}_t)(\bar{q}_t-\bar{q}_t^2-\sigma^2)-(\bar{q}_t-\bar{q}_t^2-\sigma^2)^2-(1-2\bar{q}_t+\bar{q}_t^2+\sigma^2)^2}{(1-\bar{q}_t)^2}$
$S(0,1)$	$\frac{2\bar{q}_t(\bar{q}_t-\bar{q}_t^2-\sigma^2)-(\bar{q}_t-\bar{q}_t^2-\sigma^2)^2-(\bar{q}_t^2+\sigma^2)^2}{\bar{q}_t^2}$
$S(1,1)$	$\frac{2\bar{q}_t(\bar{q}_t^2+\sigma^2)-(\bar{q}_t-\bar{q}_t^2-\sigma^2)^2-(\bar{q}_t^2+\sigma^2)^2}{\bar{q}_t^2}$

Figure 2.2: Side-payments for reputation reports, depending on the advertised QoS ($\bar{q}_t$) and noise (σ^2).

The payment scheme is the following:

- all reports submitted during the same period of time are attributed a unique sequence number, $i \in \{0, \ldots N\}$. N is the total number of collected reports (in a period).
- the feedback r_i is compared against feedback r_{i+1}, and is paid $S(r_{i+1}, r_i)$ defined according to Fig. 2.2:

The side payments depend on (a) the advertised QoS, and (b) on the variance $\sigma^2 = \sigma_n^2 + \sigma_r^2$ of the observed QoS. The first is specified in the SLA. The second can be approximated by the reputation mechanism from the reputation record of the provider (e.g. the reputation R_i is a noisy approximation of the same intended QoS). The side payments are computed and made public by the reputation mechanism at the beginning of each time period.

To prove that rational clients have the incentive to tell the truth we have to consider their beliefs. Given the SLA $(\bar{q}_t, p_t, \lambda_t)$, every client believes that the actual QoS is normally distributed around $\bar{q}_t$ with variance σ_n^2. Having observed a successful service or a failure, the client updates her prior beliefs (described by the pdf[3] $f(q)$) according to Bayes' Law into the posterior pdfs: $f(q|1)$, respectively $f(q|0)$:

$$f(q|1) = \frac{Pr[1|q] \cdot f(q)}{\int_{\mathcal{Q}} Pr[1|q] f(q) dq}; \qquad f(q|0) = \frac{(1 - Pr[1|q]) \cdot f(q)}{1 - \int_{\mathcal{Q}} Pr[1|q] f(q) dq};$$

where $Pr[1|q]$ is the probability of observing 1 given a service with quality q, and $\int_{\mathcal{Q}} Pr[1|q] f(q) dq = \bar{q}_t$ is the overall probability of observing high quality. Consequently, the *likelihood* assigned by the client to the next client's rating is described by:

$$\begin{aligned} Pr[r_{i+1} = 1 | r_i = 1] &= \int_{\mathcal{Q}} Pr[1|q] f(q|1) dq = \frac{\bar{q}_t^2 + \sigma^2}{\bar{q}_t}; \\ Pr[r_{i+1} = 1 | r_i = 0] &= \int_{\mathcal{Q}} Pr[1|q] f(q|0) dq = \frac{\bar{q}_t - \bar{q}_t^2 - \sigma^2}{1 - \bar{q}_t}; \end{aligned} \tag{2.3}$$

[3] probability distribution funtion

It is easy to verify that $Pr[1|1]S(1,1) + Pr[0|1]S(0,1) \geq Pr[1|1]S(1,0) + Pr[0|1]S(0,0)$ and $Pr[1|0]S(1,0) + Pr[0|0]S(0,0) \geq Pr[1|0]S(1,1) + Pr[0|0]S(0,1)$. In other words, when the next client reports the truth, the expected payment of a true report is always greater than the expected payment of a false report. This makes truthful reporting a Nash equilibrium. The side payments can be scaled to be always positive and budget balanced (details in [13]).

Every negative report decreases the price a client has to pay by $\lambda(\bar{q}_t, R_t - 1/N) - \lambda_t(\bar{q}_t, R_t)$. The client cannot benefit from submitting a false negative report if the loss due to lying outweighs the price cut. This can be achieved by multiplying the values in Fig. 2.2 with the constant[4]:

$$M = \frac{\lambda_t(\bar{q}_t, R_t - 1/N) - \lambda_t(\bar{q}_t, R_t)}{E(1,1) - E(0,1)} \tag{2.4}$$

where $E(r_i, o_i)$ denotes the expected payment of client i given that she has observed $o_i \in \{0,1\}$ and reports $r_i \in \{0,1\}$.

2.6 Enforcing the Truthful Reporting Strategy

The truthful equilibrium defined above is unfortunately not unique. Clients, for example, can always report negative feedback without suffering side payment losses (i.e. always reporting 0 is also a Nash equilibrium strategy). In [8] we suggest the use of *trusted reports* in order to eliminate such *undesired* equilibrium strategies. Trusted reports can be obtained from specialized agents hired to test the service of a provider.

The truthful equilibrium becomes unique when the feedback from clients is rated (as explained in the previous section) only against trusted reports. It is desirable, however, to minimize the number of trusted reports needed in order to enforce the uniqueness of the truthful equilibrium.

We modify the rating scheme from Section 2.5 such that all client reports are rated against one trusted report, randomly chosen from a small set of available trusted reports. In the extreme case the set could contain only one report; however, the right tradeoff between robustness (against the mistakes of specialized agents) and cost can be achieved by having several trusted reports.

In [8] we show that it is not necessary to have trusted reports for every time period. Using the side-payments defined above, we conclude that the truthful reporting equilibrium is very stable. It takes a big proportion (e.g. 20%) of lying agents in order to shift the reporting equilibrium, and make it rational for the other agents to lie as well. As a consequence, trusted reports need only be used in the first periods of time in order to coordinate the clients on the truthful equilibrium. Once the truthful strategy is enforced, the market

[4] multiplication or addition with a constant does not influence the truthful reporting Nash equilibrium of the side payment mechanism.

can do a passive monitoring of the reporting strategy and buy new trusted reports only when a deviation is observed. In this way, the overall number of trusted reports needed by the market becomes insignificant.

2.7 Collusion

Collusion happens when two or more clients conspire to artificially decrease the reputation of a provider, and thus decrease the price they have to pay for the service. The reputation side-payments do not make it interesting for one client to submit negative feedback, however, when several clients form a coalition and adopt a negative reporting strategy, the price-cut is cumulative and every agent benefits from the action of the group.

The use of trusted reports (as described in Section 2.6) also deters collusion. When clients are self-interested and external punishments cannot be inflicted on them, we prove that any feedback-reporting coalition is unstable, and hence, irrational.

Proposition 2 *The reputation-based Service Level Agreements are feedback-reporting collusion proof.*

PROOF.

The intuition behind this proof is that any coalition of clients (colluding to submit false feedback) is unstable. As member of such a coalition, a rational client finds it more profitable to report the truth rather than stick to the colluding strategy. Clients are free to maximize their revenue, so they will quit the coalition and choose to report truthfully.

Formally, take a subset of clients colluding on a lying strategy, and let the client c, part of the coalition, be expected to lie when submitting feedback. Client c exists, since otherwise all colluding agents report the truth. c can stick to the colluding strategy and lie: she thus benefits from the advantages of collusion, however, expects a loss due to reputation side-payments. On the other hand, c can deviate and report the truth: she thus optimizes her expected payment from the reputation mechanism but the result of collusion is less effective.

The side-payments multiplied by the factor in Equation (2.4) guarantee that the loss in reputation payment is always greater than the price-cut obtained from one false report. Therefore, it is rational for c to leave the coalition. The same argument can be applied to any colluding client; hence feedback-reporting collusion is not rational. □

Please note that stronger forms of collusion are still possible. If one client controls multiple online identities (the sybil attack) she can coordinate false reporting in order to decrease the price of service. This type of collusion should be addressed by security and social mechanisms that closely connect online and physical identity.

2.8 Experimental Evaluation

The use of reputation information greatly reduces the independent monitoring required by markets of web services. In this section we compare the mechanism described in this chapter (mechanism A) with an alternative mechanism (mechanism B) where the market only uses trusted reports (i.e. independent monitoring) to compute the penalty to service providers for QoS degradation.

We first investigate the quality of monitoring of the two mechanisms. The precision of the monitored QoS value directly impacts the revenue of service providers. When the monitored QoS value is exactly equal to the delivered QoS, service providers do not have to pay any penalty and thus obtain their maximum payoff. However, practical monitoring schemes always provide noise approximations of the delivered QoS. The noise thus introduced, translates into a non-zero expected penalty that decreases the total utility of service providers. The poorer the approximation offered by the monitoring system, the greater the utility loss of service providers.

The second criterion we employ is the monitoring cost required by the two mechanisms. While general analytical results can be obtained, we believe it is more informative to compare the two mechanisms on a realistic (however simplified) example.

Consider a web service providing closing stock quotes. A reputation-based SLA is advertised every morning and specifies the price of service, the QoS (e.g. the quote is obtained within 5 minutes of the closing time with probability $\bar{q}$) and the penalty function λ. Interested clients request the service, and then wait the answers from the service provider. They experience high quality if the answers is received before the deadline (i.e. 5 minutes after the closing time) or low quality if the answer is late or not received.

The probability of successfully answering the clients' requests depends on the available infrastructure and on the number of accepted requests. For a given provider, Fig. 2.3 plots the relation (experimentally determined) between the expected QoS (i.e. $\phi(n)$), and the number of accepted requests. The QoS actually provided to the clients is normally distributed around $\phi(n)$ with variance σ_n^2.

We assume that the closing stock quotes represent mission-critical information for the clients present in the market. Late or absent information attracts supplementary planning costs and lost opportunities. Therefore, the market price function, (i.e. $u(q)$) is assumed convex, corresponding to risk-averse clients. When $\bar{q}$ is the advertised QoS, n is the number of accepted requests, $\hat{q}$ is the QoS perceived by the market, and C denotes the fixed costs, the expected revenue of the provider is:

$$V(n, \bar{q}) = E_{\hat{q}}\Big[n \cdot \big(u(\hat{q}) - \lambda(\bar{q}, \hat{q})\big) - C\Big];$$

By using the mechanism A, the market perceives a QoS equal to: $\hat{q}_A = \phi(n) + \eta_n + \eta_r$ where η_r is the noise introduced by reporting mistakes, normally distributed around 0 with variance σ_r^2. For a price function $u(q) = q^2$, the fixed cost $C = 100$, the standard

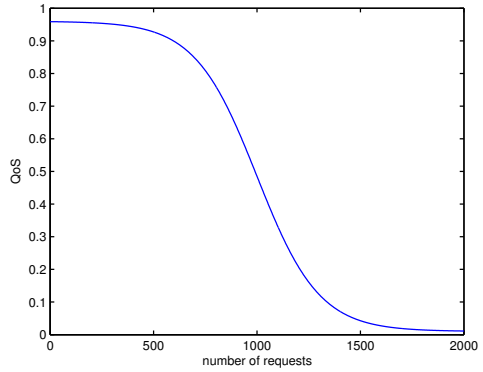

Figure 2.3: The QoS as a function of the number of requests accepted by a provider. (Experimentally determined)

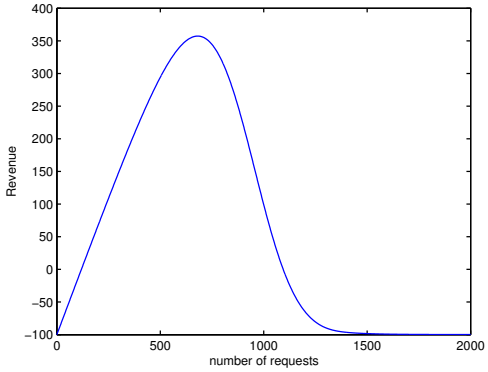

Figure 2.4: The revenue function of the provider depending on the number of accepted requests.

deviations $\sigma_n = 3\%$, $\sigma_r = 4\%$, and a penalty function $\lambda(\bar{q}, \hat{q}) = 2\big(p(\bar{q}) - p(\hat{q})\big)$, Fig. 2.4 shows the optimal revenue of the provider as a function of n. The optimal value of the payoff function is reached for $n_t = 681$, when $\bar{q} = 0.858 = \phi(681)$, as predicted by Proposition 1. Mechanism B satisfies the same optimality and incentive-compatible properties for the service provider. Different price functions or quality functions generate different optimal parameters, however, they do not modify the qualitative properties of the mechanism: providers deliver at least their declared QoS, and clients have the incentives to report the truth.

The average, per-client, utility loss of a service provider is defined as the expected penalty a provider has to pay as a consequence of an inaccurate approximation of the delivered QoS (as computed by the monitoring mechanisms). When $\hat{q}_A$ and $\hat{q}_b$ are the monitored QoS values provided by the two mechanisms, the utility losses caused by the two mechanisms are:

$$UtilLoss_A = E_{\hat{q}_A}\big[\lambda(\bar{q}, \hat{q}_A)\big]; \qquad UtilLoss_B = E_{\hat{q}_B}\big[\lambda(\bar{q}, \hat{q}_B)\big];$$

computed at the optimal QoS, $\bar{q}$. A higher variance of $\hat{q}$ increases the utility losses of providers. Typically, mechanism B has less information than mechanism A about the delivered QoS and therefore generates higher losses for providers. The difference in the average utility loss per client generated by the two mechanisms is shown in Fig. 2.5, as a function of the number of trusted reports employed by mechanism B. To reach the same performance, mechanism B needs approximately 75 trusted reports, i.e. 11% of the number of service requests.

The administrative costs of the mechanism A consist of (a) the reputation side-payments and (b) the cost of trusted reports. The cost of mechanism B consists only of trusted reports. The cost of a trusted report is assumed equal to $(1 + \delta)$ times the price of service (e.g. the monitoring agent buys the service and receives a commission δ). We take $\delta = 0.1$.

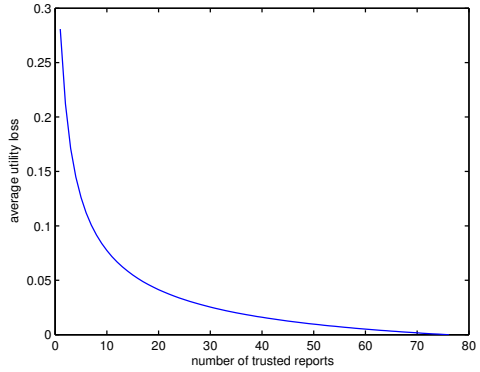

Figure 2.5: The difference in client utility loss caused by using only trusted reports.

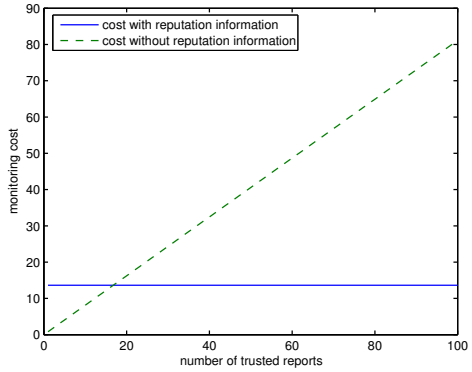

Figure 2.6: The monitoring cost of not using reputation information.

For the same parameter values as above, the reputation side-payments given in Fig. 2.2 (properly scaled to be positive and multiplied with the correction factor defined by Equation 2.4) become: $S(1,1) = 2.3\%$, $S(0,1) = 0$, $S(1,0) = 1.6\%$ and $S(0,0) = 1.7\%$ of the price of the perfect service (i.e. $u(1)$). Fig. 2.6 plots the difference in monitoring costs between the mechanisms A and B for different number of trusted reports employed by mechanism B. For similar performance (i.e. 75 trusted reports) mechanism B has monitoring costs that are 4 times higher.

Please note that the utility loss in Fig. 2.5 is for every client. When mechanisms A and B have the same monitoring cost (i.e. mechanism B uses approximately 20 trusted reports) a service provider looses on the average approx. 4.5% more utility for every customer as a consequence of not using reputation-based monitoring. This apparently insignificant amount, multiplied by the number of total clients (i.e. 681), generates significant losses for the provider.

2.9 Related Work

Our work can best be situated at the confluence of two lines of research in service-oriented computing: electronic contract enforcement and reputation-based selection of services.

The legal system is seen as inappropriate for e-commerce disputes [2] and therefore alternative dispute resolution mechanisms have been proposed to avoid the escalation of disputes to the legal stage. Electronic contract enforcement covers both non-discretionary approaches (e.g. preventive security mechanisms) as well as discretionary ones (e.g. different control mechanisms that are applied when contract rules are breached). Concrete progress has been made in the areas of e-contract formal models ([20], [19]), contract performance monitoring([20], [14], [11]), mediation of services through trusted third parties ([16], [17]) and security infrastructures for safe service delivery([6], [5]).

Reputation mechanisms have emerged as efficient tools for service discovery and selection [18]. When electronic contracts cannot be enforced, users can protect themselves against cheating providers by looking at past behavior (i.e. the provider's reputation). Lie et al. [10] present a QoS-based selection model that takes into account the feedback from users as well as other business related criteria. The model is extensible and dynamic. In the same spirit, [9] proposes *verity*, a QoS measure that takes into account both reputation and the terms of the SLA. [12] and [1] propose concrete frameworks for service selection based on provider reputation.

An interesting approach is proposed by Deora et al. in [4]. The authors argue that the expectations of a client greatly influence the submitted feedback, and therefore both should be used when assessing the QoS of a provider.

Our work is novel in three main aspects. First, client feedback becomes a first-class citizen of the interaction model. Reputation has a clear semantics and is used to compute monetary penalties for deviations from the advertised QoS. This makes it possible to rigourously analyze the strategies of rational service providers and give theoretical proofs regarding the properties of the mechanism: e.g. truthful declaration of QoS, low monitoring cost. Second, our model is free from any probabilistic assumptions about the behavior of clients and providers. Clients and providers are assumed to be self interested and free to maximize their revenues. Third, we present a practical mechanism for ensuring truthful feedback from clients that also deters collusion.

2.10 Conclusion

Without proper monitoring of the delivered QoS, self-interested providers have the incentive to cheat by promising a higher than intended QoS. In this chapter we present a new form of SLAs where the final price paid by clients depends on the actual quality delivered by the service provider, as computed by a reputation mechanism. When clients honestly submit feedback, a reputation mechanism is efficient in monitoring the real QoS and makes it rational for all service providers to keep their promises.

As a second contribution we show how a side-payment scheme can be used in a market of web services to elicit honest feedback from rational clients. Moreover, a small number of trusted reports can prevent collusion and enforce truth-telling as a unique strategy. We proved also that only few trusted reports are temporarily needed in order to coordinate the clients on the truthful strategy. After this initial phase, the truthful strategy is quite stable (i.e. it takes a large group of agents to change the reporting strategy of the whole community) and the market should only assume a passive, monitoring role. Our mechanism therefore generates significantly lower cost than traditional monitoring mechanisms.

We thus describe a simple, robust mechanism that eliminates incentives for selfish providers to cheat, at a much lower cost. The assumptions behind the mechanism are fairly general, making it a candidate for many practical settings.

Chapter 3

Decentralized Orchestration of Composite Web Services

Traditional, centralized orchestration of composite web services often leads to inefficient routing of messages. To solve this problem, we present a novel scheme to execute composite web services in a fully decentralized way. We introduce service invocation triggers, a lightweight infrastructure that routes messages directly from the producing service to the consuming one, enabling fully decentralized orchestration. We will measure total network traffic, comparing centralized workflow orchestration versus decentralized orch.

This chapter will be completed in the second version of this deliverable.

Chapter 4

Conclusion

Service-oriented computing enables the construction of distributed applications by integrating services that are available over the web [15]. The acceptance of this service-oriented environment cannot work without a trustworthiness of its agents and it also requieres efficient approaches for communicating.

We have described a simple, robust mechanism that eliminates incentives for selfish providers and clients to cheat. The assumptions behind the mechanism are fairly general, making it a candidate for many practical settings.

In the second version of this deliverable we will describe also schemes for executing composite web services communicating in an efficient decentralized way.

Bibliography

[1] B. Alunkal, I. Veljkovic, G. Laszewski, and K. Amin. Reputation-Based Grid Resource Selection. In *Proceedings of AGridM*, 2003.

[2] A. Carblanc. Privacy protection and redress in the online environment: Fostering effective alternative dispute resolution. In *In Proceedings of the 22nd International Conference on Privacy and Personal Data Protection*, Venice, 2000.

[3] A. Dan, D. Davis, R. Kearney, A. Keller, R. King, D. Kuebler, H. Ludwig, M. Polan, M. Spreitzer, and A. Youseff. Web services on demand: WSLA-driven automated management. *IBM Systems Journal*, 43(1):136–158, 2004.

[4] V. Deora, J. Shao, W. Gray, and J. Fiddian. A Quality of Service Management Framework Based on User Expectations. In *Proceedings of ICSOC*, 2003.

[5] R. Handorean and G. Roman. A framework for requirements monitoring of service based systems. In *Proceedings of ICSOC*, 2003.

[6] Y.-J. Hu. Trusted Agent-Mediated E-Commerce Transaction Services via Digital Certificate Management. *Electronic Commerce Research*, 3, 2003.

[7] R. Jurca and B. Faltings. An Incentive-Compatible Reputation Mechanism. In *Proceedings of the IEEE Conference on E-Commerce*, Newport Beach, CA, USA, 2003.

[8] R. Jurca and B. Faltings. Enforcing Truthful Strategies in Incentive Compatible Reputation Mechanisms. In *Proceedings of the Workshop on Internet and Network Economics (WINE)*, Hong Kong, China, 2005.

[9] S. Kalepu, S. Krishnaswamy, and S. Loke. Verity; A QoS Metric for Selecting Web Services and Providers. In *Proceedings of WISEW*, 2003.

[10] Y. Liu, A. Ngu, and L. Yeng. QoS Computation and Policing in Dynamic Web Service Selection. In *Proceedings of WWW*, 2004.

[11] K. Mahbub and G. Spanoudakis. A framework for requirements monitoring of service based systems. In *Proceedings of ICSOC*, 2004.

[12] E. M. Maximilien and M. P. Singh. Toward Autonomic Web Services Trust and Selection. In *Proceedings of ICSOC*, 2004.

[13] N. Miller, P. Resnick, and R. Zeckhauser. Eliciting Informative Feedback: The Peer-Prediction Method. Forthcoming in Management Science, 2005.

[14] Z. Milosevic and G. Dromey. On expressing and monitoring behaviour in contracts. In *Proceedings of EDOC*, Lausanne, Switzerland, 2002.

[15] M. P. Papazoglou and D. Georgakopoulos. Introduction: Service-oriented computing. *Communications of the ACM*, 46(10):24–28, Oct. 2003.

[16] G. Piccinelli, C. Stefanelli, and D. Trastour. Trusted Mediation for E-service Provision in Electronic Marketplaces. *Lecture Notes in Computer Science*, 2232:39, 2001.

[17] R. Shuping. A Model for Web Service Discovery with QoS. *ACM SIGecom Exchanges*, 4(1):1–10, 2003.

[18] M. P. Singh and M. N. Huhns. *Service-Oriented Computing*. Wiley, 2005.

[19] Y.-H. Tan and W. Thoen. A Logical Model of Directed Obligations and Permissions to Support Electronic Contracting. *International Journal of Electronic Commerce*, 3(2), 1999.

[20] L. Xu and M. A. Jeusfeld. Pro-active Monitoring of Electronic Contracts. *Lecture Notes in Computer Science*, 2681:584–600, 2003.