A WS-Agreement Extension for Specifying Temporal Properties in SLAs *

Carlos Müller, Octavio Martín-Díaz, Manuel Resinas,
Pablo Fernández y Antonio Ruiz-Cortés

Departamento de Lenguajes y Sistemas Informáticos

Escuela Técnica Superior de Ingeniería Informática

Universidad de Sevilla

41012 Sevilla

Abstract

Service level agreements (SLA) in service oriented architectures allow to regulate the service trading from providers to consumers. So SLAs must describe the agreement properties in a precise manner. WS-Agreement proposes a standard for describing SLAs to specify "which" service will offer and "how" it is offered. However there is another important question about services which is "when". Temporality affects orthogonally to all aspects of a SLA (i.e. *the entire SLA, functional or non-functional properties*) and it is necessary to express more precisely the SLAs. WS-Agreement identifies the necessity of specifying temporality on agreement terms but it does not describe how temporality must be expressed and, to the best of our knowledge, there is no extension of WS-Agreement that deals with that problem. Therefore we propose to extend the standard with a temporal XML schema to describe several validity periods that could be disjoint, non-disjoint, and/or periodical.

1 Introduction

Service oriented architectures are based on the use of loosely coupled service to support the requirements of business processes and users. In this context, service level agreements (SLAs) [12, 11, 19] can be used to regulate the execution of the services and to give guarantees related to them.

A SLA usually specifies "*which*" service is offered and "*how*" it is offered. That is to say, it includes requirements and guarantees about functional, and non-functional properties of the services. However, another important question about services is "*when*". Temporality affects orthogonally to all aspects of a SLA because it may refer to the entire agreement (e.g. *the agreement expires on 2007/05/31*); to any functional property of the service (e.g. *this operation of the service is available from 8:00h to 18:00h*), or to any non-functional property that appears in the SLA (e.g. *the response time is 30 ms from 8:00h to 17:00h and 15 ms from 17:00h to 8:00h*). Therefore, a SLA with more temporality allows us to express precisely the periods of time in which its terms are valid.

The most significant language to specify SLAs is WS-Agreement [11]. WS-Agreement is a specification that provides a schema for defining SLAs and a protocol for creating them based on a mechanism of templates. However, WS-Agreement only defines the general structure of the agreement and other parts such as defining domain-specific expressions or specific condition expression languages are out of the scope of the specification. This is also the case

*This work has been partially supported by the European Commission (FEDER) and Spanish Government under CICYT project Web-Factories (TIN2006-00472).

of temporality: WS-Agreement identifies that it is necessary to include temporality in the terms of the agreement, but it does not establish how to specify it. Furthermore, to the best of our knowledge, there is no extension to WS-Agreement that tackles the problem of temporality.

In this paper, we propose an extension to give WS-Agreement full support to temporality. To define it, we build on a previous work [17], in which we presented operational semantics on a constraint-based temporal-aware matchmaking. In this paper, we define a *temporal XML schema* based on that work and we describe how this temporal schema can be applied to the different elements of WS-Agreement.

The advantages of our approach are the following: (i) we apply temporality not only to the entire agreement and the agreement terms but also to other elements of WS-Agreement such as the creation constraints, which are used to create agreements based on templates, and business values, which are used to express preferences about the terms of the agreement; (ii) we support expressive specifications of validity periods such as composed validity periods like "*From 8:00h to 14:00h and From 16:00h to 18:00h*" and periodical validity periods like "*From Mondays to Fridays, from 8:00 to 18:00*".

This paper is structured as follows. Section 2 introduces a case of study in which temporality is an important feature. Section 3 introduces the SLA structure for WS-Agreement and its temporal properties with previous scenario. Section 4 exposes our proposal of WS-Agreement extension. Section 5 compare de related temporality works. And Section 6 exposes our conclusions and future works.

2 A Case of Study

In general, temporal issues are present in the majority of agreements in real-world scenarios. In this section we explore a particular case where a provider offers computing services to other organisations; i.e. customers send jobs (data to be processed by a certain algorithm) to be executed in the provider's infrastructure. This specific scenario represent a common situation in research fields with intensive computational requirements [14, 3] having a wide set of temporal aspects that exemplify most of our model.

In this scenario, a provider is likely to be looking for an optimisation in the usage of its resources; that means unused (or underused) resources represent a lack of benefits and, therefore, a low recovery of the initial investment. In doing so, agreement offers should vary in a certain period based on the actual resource usage in the period.

As an example we stand a provider offering computing services with the following features:

- The validity of SLA is from october 1/2007 to december, 30/2007.
- All Sundays at 23:00 h. servers are down for an hour due to maintenance tasks.
- Provider needs the 20% of its resources for its own computing necessities from Mondays to Fridays, from 8:00 to 18:00.
- Rest of time, 100% of server resources can be offered to consumers.
- A client that wants to use the service must specify in his agreement offer: a request; an algorithm for processing the request; the dedication of the server expressed in percentage of resources; the time of the request; and at last the temporal execution pattern for the request (that means an estimation about when are going to be the service invocations).

In the case of study we consider a consumer that needs to compare two different algorithms with the same requirements on computational resources, so resources must be shared between them.

3 WS-Agreement in a Nutshell

WS-Agreement specifies an XML-based language and a protocol for advertising the capabilities of service providers, creating agreements based on agreements offers, and for

monitoring agreement compliance at runtime. The interaction protocol proposed for WS-Agreement is as follows. An initiator of the agreement process asks for agreement templates to an agreement responder. The initiator sends to the responder an agreement offer taking into account the agreement variability allowed by the responder defined in the template. Then the responder accepts or rejects the agreement offer. If the responder rejects, the process may start again.

An agreement offer is composed of:

Name: it identifies the agreement and it can be used for reference from other agreements.

Context: it includes information such as the name of parties and their roles of initiator or responder of the agreement; It identifies the agreement template if needed; the agreement lifetime is defined by means of an element called "ExpirationTime"; and any other agreement information could be included under the "Any" element.

Terms: agreement terms are wrapped by a term compositor, which allows simple terms or sets of terms be denoted with "ExactlyOne", "OneOrMore", or "All". There are two types of terms: (1) *Service Description Terms*: they provide information to instantiate or identify services involved in the agreement, and information about the measurable service properties that are needed in the agreement creation process. (2) *Guarantee Terms*: they describe the service levels agreed by the parties. They comprise a Service Level Objective (SLO) specified as a target for a key performance indicator, or as a "CustomServiceLevel" element in a customized way; the scope of the term; a "QualifyingCondition" that specifies the conditions under which the term is applied; and information about bussiness values as preference, importance, penalty or reward of the guarantee term.

WS-Agreement allows to specify templates with the same structure than agreement offers but including agreement creation constraints that have to be taken into account in agreement creation process. These constraints describe the agreement variability allowed by a party; they could be denoted as general "Constraints", or "Items" pointing to specific locations with their own constraint (grouped or not). Any other way is also allowed by the "Any" element.

Regarding with temporal properties, WS-Agreement states that the lifetime for the entire agreement must be included in Context into "ExpirationTime" element as a final instant. It is also recommended the use of "QualifyingCondition" element in Guarantee Terms for describing temporal conditions about terms. However they do not specify the way these temporal conditions must be exposed.

The temporal execution pattern described in the previous scenario is too complex for WS-Agreement because the example includes several temporal expressions which cannot be defined in WS-Agreement. In order to describe this example with this standard, we have to reduce it as denoted in Figure 1, specifying only an expiration time (not initial time); the request; the algorithm; the dedication of the server; and the temporal execution pattern for the request. The latter has to be changed with a simple value of execution time. The necessity of restarting server periodically has to be removed from the example and the possible usage of server corresponds to the worst case (80%). Therefore resources of our example are shared (40% for each request) and the provider guarantees a maximun of 80% of overall server usage.

The provider of the example can describe an agreement template considering its necessities of processing and restarting as creation constraints for the agreement. So consumers have to take these constraints into acount in their agreement offers. But again, we find that WS-Agreement does not allow to describe that temporal expressiveness so we have to reduce the example restricting only the algorithms allowed, and the possible range of execution time in hours too. Figure 2 depicts the structure of an agreement template, which is similar to Figure 1 but excluding concrete values of the service features and including creation constraints at the end of the template.

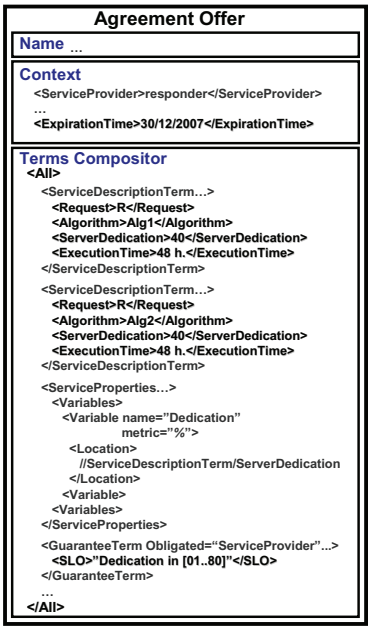

Figure 1: Example of Agreement Offer.

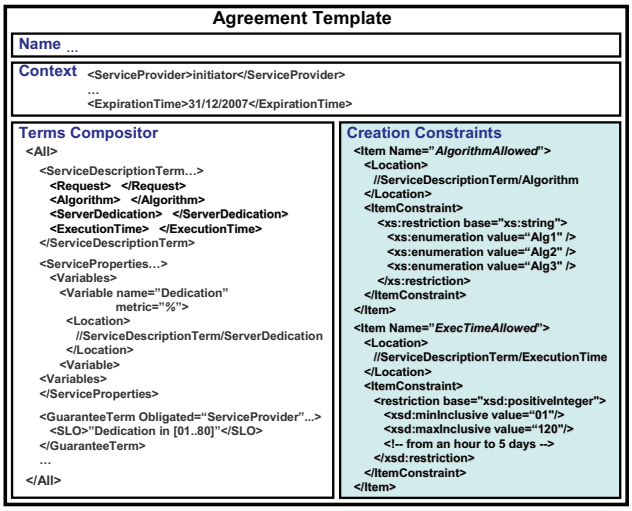

Figure 2: Example of Agreement Template.

4 Our Proposal

We propose a WS-Agreement general extension for describing temporal properties in SLA. First we must specify a temporal schema as general as possible for including several forms of validity periods.

4.1 Temporal Schema

In a previous work we made a study about temporal expressiveness for web services [18].

The validity periods over SLAs may be composed of one or more intervals, that can be also periodic. There are several types of intervals, namely non-disjoint, disjoint (both mentioned by Allen in [2]), and/or periodical. A non-disjoint interval is composed of a single interval. A disjoint interval is composed of several sub-intervals, so that it does not comprise all time points between its lower and upper ends. And an interval is periodical if it is repeated regularly.

We have designed an XML schema for describing these periods in practice. An interval is the basic element; different non-disjoint intervals can be grouped together so that more complex intervals can be composed. Several authors [16, 4] propose a more friendly representation of an XML Schema with UML class diagrams. So Figure 3 represents our "twsag.xsd" in an UML class diagram with three interfaces denoting the three types above mentioned: (1) **Interval**: it stands for the basic element; it is comprised of an initial time and a duration expressed in seconds, hours, days, or months; (2) **Disjoint**: it stands for disjoint intervals constituted of a set of intervals related by a logic operator (or, and, or xor); (3) **Periodical**: it stands for periodic intervals, be either disjoint or non-disjoint. Its periodicity is comprised of the number of period repetitions and a frequency expressed in seconds, hours, days, or months, which denotes the time between two consecutive intervals.

4.2 WS-Agreement Temporal-Extension

Depending on how the validity periods affect to the agreement terms, we can classify them in two groups: (1) global periods (GP) if validity periods wrap all agreement terms; and (2) local periods (LP) in other cases. We have

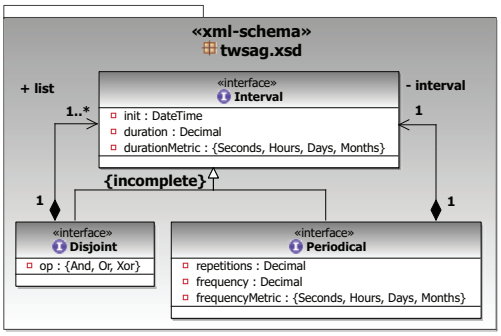

Figure 3: Schema for Temporal Intervals.

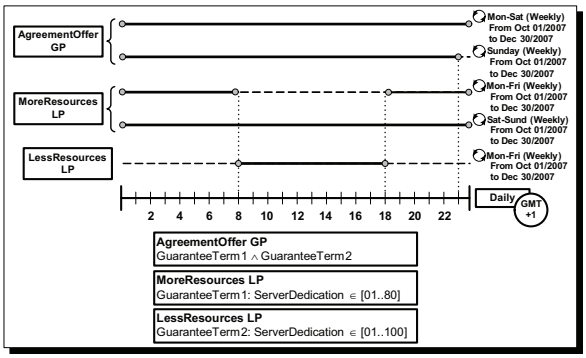

Figure 4: Validity periods for the case of study.

study the inclusion of these types of periods in the WS-Agreement structure. And we must study where to include these types of validity periods in the WS-Agreement structure.

WS-Agreement specifies the lifetime of agreements by means of an "Expiration Time" in the context. Thus, it only allows a non-disjoint GP, starting from the current date. For a lifetime to be expressed without restrictions, we propose to include a new element called "GlobalPeriod" in order to describe it as an "Interval" element of our temporal schema. This solution is compatible with WS-Agreement structure because we make use of the "Any" element which allows to include any information in the context.

WS-Agreement recommends to specify temporality regarding with agreement terms in "QualifyingCondition" element. Thus, we propose to specify the LP in this element described as an "Interval" element of our temporal schema.

Figure 4 denotes the global and local periods for the scenario described in Section 2. Figure 5 shows an offer using our WS-Agreement extension for describing the validity periods in such case of study. Note that non-disjoint intervals are put into a single periodical non-disjoint interval which conforms the agreement offer GP; and periodical disjoint intervals are used to conform the agreement offer LPs.

It is important to remark that WS-Agreement only includes temporal properties about terms in guarantee terms. But in our example functional agreement properties described in the service description terms are only active at specific periods (e.g. *we must use the service description term with 80% of dedication server, in case of periods with a maximun of 80% of dedication server allowed*). So we make use of the term compositor to wrap the affected service term and the guarantee term which contains the desired validity period.

We also allow to specify temporal properties regarding with agreement creation constraints. There are two ways to describe them: either to allow validity periods on single constraints (e.g. *"Provider must allow execution tests with a maximum of 40% of server dedication, 48 hours before agreement initiation date"*); or to allow several constraints apart from the validity period definition (e.g. the previous constraints without validity period: *"Provider must allow execution tests with a 40% of server dedication"*, and also *"Provider must assure a maximum execution time of 24 hours"*, both active during the validity period: *"48 hours before agreement initiation date"*). For temporality in creation constraints to be allowed, we propose to describe them as an "Interval" element of our temporal schema: (1) a new element under the "Item" of creation constraints, for describing temporal periods on a single constraint (by means of "Any" element of WS-Agreement); and (2) the "Constraint"

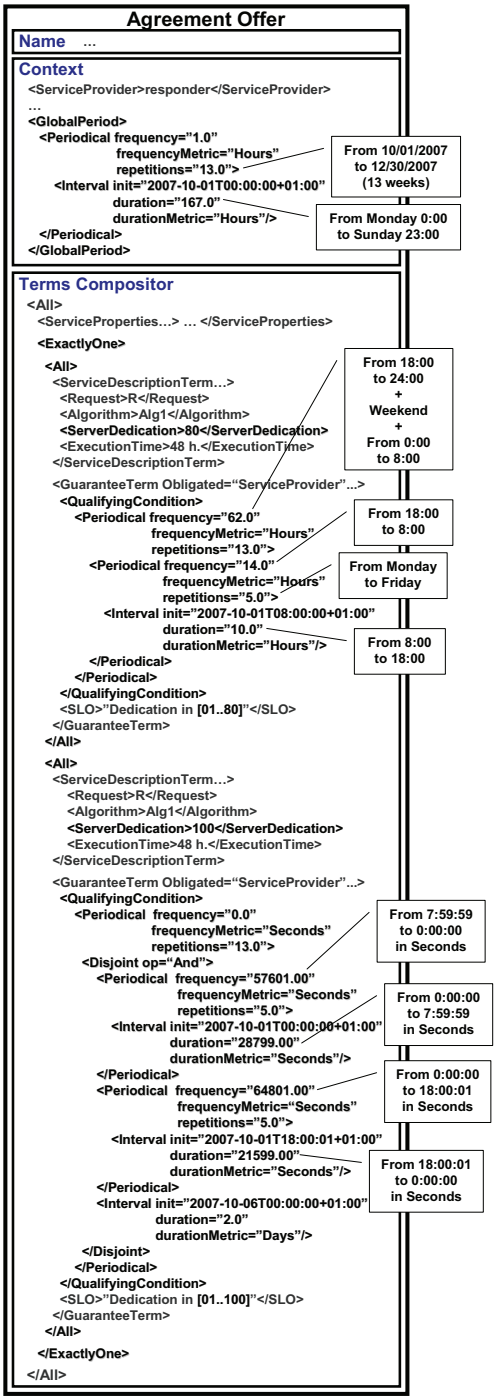

Figure 5: Offer using our WS-Ag. Extension.

```
<template...>
...
  <CreationConstraints>
    <Item Name="TestPrevious">
      <Location>
        //ServiceDescriptionTerm/ServerDedication
      </Location>
      <ItemConstraint>
        <xsd:restriction base="xsd:positiveInteger">
          <xsd:minInclusive value="01"/>
          <xsd:maxInclusive value="40"/>
        </xsd:restriction>
      </ItemConstraint>
      <Interval init="2007-09-29T00:00:00+01:00"
                duration="2.0"
                durationMetric="Days"/>
    </Item>
  </CreationConstraints>
  ...
</template>
```

Figure 6: Creation Constraints with Temporality.

element for temporal periods on several constraints. Figure 6 denotes this situation on single constraints, applied to an example about testing requests before an agreement initiation date.

5 Related Work

Several authors have studied temporality in their works. In Table 1 we show a comparative of their temporality, putting proposals using traditional web at left side and proposals using semantic web at right.

The table denotes that authors who regard GPs, only mention "Non-Periodical" and "Non-Disjoint" intervals, but neither "Periodical" nor "Disjoint" intervals. And authors who regard LPs, mention "Periodical" and "Non-Disjoint" and only WSMO/WSML and METEOR-S show interest for regarding in future "Non-Periodical" intervals in LPs. Rest of authors neither mention "Non-Periodical" nor "Disjoint" intervals in their works. The reason for that lack of "Disjoint" intervals in works may be that it can be expressed with several "Non-Dijoint" intervals. But it is less expressive than using all types of interval which we state.

The more complete proposals are WSML(HP) and WSOL because they regard GPs and LPs in their works. Other

Proposals	Lodi et al. [14]	WS-QoS [23]	EWSDL [5]	UDDIe [1]	WSML (HP) [22]	WSOL [24]	WSLA [15]	Gouscos et al. [10]	Trastour et al. [25]	DAML-QoS [6]	Gonzalez. et al. [9]	Li & Horrocks [13]	QoSOnt [8]	WSMO/WSML [7]	METEOR-S [20]
GP/NP	✓	✓	✓	✓	✓	✓			✓	✓	✓	✓	∼	∼	∼
GP/P															
GP/ND	✓	✓	✓	✓	✓	✓			✓	✓	✓	✓	∼	∼	∼
GP/D															
LP/NP													∼	∼	∼
LP/P					✓	✓	✓	✓							
LP/ND					✓	✓	✓	✓					∼	∼	∼
LP/D															

D=Disjoint, ND=Non-Disjoint, P=Periodical, NP=Non-Periodical.

Table 1: Traditional & semantic web proposals

proposals like QoSOnt, WSMO/WSML, and METEOR-S have declared that they will study on GPs and LPs in future works.

6 Conclusions and Future Work

In this paper, we present an extension to give full temporality support to WS-Agreement. First, we describe a schema to define expressive validity periods which are composed of several intervals or periodical intervals. Then, we present how these validity periods can be used to extend WS-Agreement for specifying temporal properties. We describe how to apply these periods over entire agreements, functional or non-functional properties, creation constraints in templates.

However, there are still several open issues that we want to solve in further research. For instance we want to validate our temporal schema in different scenarios; we want to develop an implementation of our extension of WS-Agreement based on an adaptation of the operational semantics on matchmaking defined in previous works [17, 21]; we want to work on compatibility with other agreement proposals like [19].

References

[1] A. S. Ali, R. Al-Ali O. Rana, and D. Walker. UDDIe: An Extended Registry for Web Services. In *Proc. of the IEEE Int'l Workshop on Service Oriented Computing: Models, Architectures and Applications at SAINT Conference*. IEEE Press, January 2003.

[2] James F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11), 1983.

[3] M. Balaziska, H. Balakrishnan, and M. Stonebraker. Contract-Based Load Management in Federated Distributed Systems. In *Proc. of the ACM Symposium on Networked Systems Design and Implementation*, San Francisco, California, March 2004. ACM.

[4] M. Bernauer, G. Kappel, and G. Kramler. Representing XML Schema in UML - A Comparison of Approaches. In *1^{st} Int.l Conf. on Web Engineering*, volume 3140 of *LNCS*, pages 440–444, Munich, Germany, 2004. Springer Verlag.

[5] Y. Chen, Z. Li, Q. Jin, and C. Wang. Study on QoS Driven Web Services Composition. In *Proceedings of the 8^{th} Asia-Pacific Web Conference*, volume 3841 of *LNCS*, pages 702–707, Harbin, China, January 2006. Springer Verlag.

[6] Zhou Chen, Chia Liang-Tien, and Lee Bu-Sung. Semantics in Service Discovery and QoS Measurement. *IT Pro - IEEE Computer Society*, pages 29–34, 2005.

[7] J. de Bruijn, C. Feier, U. Keller, R. Lara, A. Polleres, and L. Predoiu. WSML Reasoning Survey, November 2005.

[8] G. Dobson and A. Sánchez-Macián. Towards Unified QoS/SLA Ontologies. In *Proc. of the 3^{rd} IEEE International ICWS/SCC Workshop on Semantic and Dynamic Web Processes*, pages 169–174, Chicago, IL, September 2006. IEEE Computer Society Press.

[9] J. González-Castillo, D. Trastour, and C. Bartolini. Description Logics for Matchmaking of Services. Technical Report HPL-2001-265, Hewlett-Packard.

[10] D. Gouscos, M. Kalikakis, and P. Georgiadis. An Approach to Modeling Web Service QoS and Provision Price. In *Proc. of the IEEE Int'l Web Services Quality Workshop (at WISE'03)*, pages 121–130.

[11] W3C Grid Resource Allocation Agreement Protocol WG (GRAAP-WG). Web Services Agreement Specification (WS-Agreement) (v.10/2006), 2006. `https://forge.gridforum.org/sf/projects/graap-wg`.

[12] IBM. Web Service Level Agreement (WSLA) Language Specification, 2003. `http://www.research.ibm.com/wsla/`.

[13] L. Li and I. Horrocks. A Software Framework for Matchmaking based on Semantic Web Technology. In *Proc. of the 12^{th} ACM Intl. Conference on World Wide Web (WWW'03)*, pages 331–339, 2003.

[14] G. Lodi, F. Panzieri, D. Rossi, and E. Turrini. SLA-Driven Clustering of QoS-Aware Application Servers. *IEEE Transactions on Software Engineering*, 33(3):186–196, 2007.

[15] H. Ludwig, A. Keller, A. Dan, and R.P. King. A Service Level Agreement Language for Dynamic Electronic Services. Technical Report RC22316 (W0201-112), IBM, 2002.

[16] E. Marcos, V. de Castro, and B. Vela. Representing Web Services with UML: A Case Study. In *1^{st} Int.l Conf. on Service-Oriented Computing*, volume 2910 of *LNCS*, pages 17–27, Trento, Italy, 2003. Springer Verlag.

[17] O. Martín-Díaz, A. Ruiz-Cortés, A. Durán, and C. Müller. An approach to temporal-aware procurement of web services. In *3rd International Conference on Service Oriented Computing (ICSOC)*, pages 170–184. LNCS 3826, 2005.

[18] Carlos Müller, Octavio Martín-Díaz, Antonio Ruiz-Cortés, and José M. García. Consistencia y conformidad en un contexto temporal. In *Métodos y Herramientas para el Desarrollo de Aplicaciones - Actas del taller de trabajo ZOCO (XI Jornadas de Ingeniería del Software y Bases de Datos)*, 2006.

[19] OASIS and UN/CEFAT. Electronic business using XML (ebXML), 2007. `http://www.ebxml.org/`.

[20] N. Oldham, K. Verma, A. Sheth, and F. Hakimpour. Semantic WS-Agreement Partner Selection. In *15^{th} International World Wide Web Conference*, pages 697–706, Edinburgh, Scotland, 2006. ACM.

[21] A. Ruiz-Cortés, O. Martín-Díaz, A. Durán, and M. Toro. Improving the Automatic Procurement of Web Services using Constraint Programming. *Int. Journal on Cooperative Information Systems*, 14(4), 2005.

[22] A. Sahai, V. Machiraju, M. Sayal, L.J. Jin, and F. Casati. Automated SLA Monitoring for Web Services. Research Report HPL-2002-191, HP Laboratories.

[23] M. Tian, A. Gramm, T. Naumowicz, H. Ritter, and J. Schiller. A Concept for QoS Integration in Web Services. In *Proc. of the IEEE Int'l Web Services Quality Workshop (at WISE'03)*, pages 149–155.

[24] V. Tosic, B. Pagurek, K. Patel, B. Esfandiari, and W. Ma. Management Applications of the Web Service Offering Language (WSOL). In *Information Systems*, pages 564–586, 2005.

[25] D. Trastour, C. Bartolini, and J. González-Castillo. A Semantic Web Approach to Service Description for Matchmaking of Services. Technical Report HPL-2001-183, HP.