Flexible SLA Negotiation Using Semantic Annotations

Henar Muñoz[1], Ioannis Kotsiopoulos[2], András Micsik[3], Bastian Koller[4], and Juan Mora[1]

[1] Telefónica Investigación y Desarrollo S.A, Valladolid, Spain
{henar,mora}@tid.es
[2] The University of Manchester, United Kingdom
ioannis@cs.man.ac.uk
[3] MTA SZTAKI, Budapest, Hungary
micsik@sztaki.hu
[4] High Performance Computing Center Stuttgart, Germany
koller@hlrs.de

Abstract. Moving towards a global market of services requires flexible infrastructures that will deal with the inevitable semantic heterogeneity that occurs during the negotiation that precedes the trading of a service. In order to reach an agreement, the negotiating parties need to understand the concepts describing the Quality of Service (QoS) terms which are part of the Service Level Agreement (SLA). The use of semantic annotations can increase the level of flexibility and automation, allowing the two parties to use their own terminology as long as it is related to the commonly understood conceptual model. This paper discusses how SLA negotiation will benefit from the use of a lightweight backwards compatible semantic annotation mechanism.

Keywords: Service Level Agreement (SLA), semantic annotations, SLA negotiation, ontologies.

1 Introduction

Since the 1980s, Service Level Agreements (SLAs) were established as tools for stating the Quality of a Service. Mainly used in the Telecommunication domain, and used as paper print-outs there was a tendency in the research community to try to adapt the SLA concepts on other domains [1]. Considerable research on SLAs and related technologies has been carried out in the Grid domain, especially in projects dealing with Virtual Organizations and Grids supporting relationships of its users.

Despite the extensive research on SLAs, the exact coverage of an SLA in terms of electronic contracts is still under debate. Some argue to use SLA as a complete replacement of contracts, but in reality the complexity and the sheer size of contracts has led us to assume that SLAs will cover statements of parts of electronic contracts defining the QoS terms, as well as the obligations on all involved business parties. The existence of SLAs is based on the assumption that an agreement can be reached. This may be simple to achieve when we consider services which can be represented on plain system parameters (e.g. lead time or number of containers in the domain of

A. Dan, F. Gittler, and F. Toumani (Eds.): ICSOC/ServiceWave 2009, LNCS 6275, pp. 165–175, 2010.

logistics) or when the business participants speak a common language. But with the involvement of different language domains the creation of SLAs gets more complex and with this increased complexity it looses the interest of business end-users.

The increase of the service market and the emergence of cloud computing requires tools that will efficiently handle the whole SLA lifecycle without the need for human interaction. One of the major obstacles in this attempt for automation is the lack of formal semantics associated with the SLA terminology. Ontologies and semantic technologies can address this obstacle to a large extent as we will demonstrate with this paper. The rest of the paper is structured as follows, Section 2 discusses related work. Section 3 presents the proposed specification, Section 4 the related ontologies and in Section 5 we explain its applicability for SLA negotiation. Section 6 provides the details about our implementation experiments. Finally, our conclusions and ideas for further work are presented in Section 7.

2 Related Work

The presented work targets mainly two different areas; SLA Negotiation protocols and Semantic Annotations of SLAs. In the following we give a short overview of identified work in these areas.

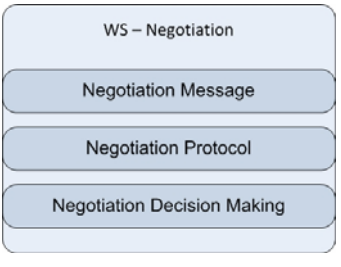

Fig. 1. Layers in SLA negotiation

With the advent of the usage of SLAs in e-business, several attempts have been made to support the creation (negotiation) of Service Level Agreements in an optimal way for all involved business entities. Hung et al [4] describe WS-Negotiation as an independent declarative XML language for Service Providers and Customers to come to an agreement. Their model was based on three layers (cf. Figure 1), the Negotiation message (describing the contents of the exchanged message), the Negotiation Protocol (describing how to exchange these messages) and the Negotiation Decision Making (describing the decision taking based on internal strategies of the entity).

Defining the Negotiation Message, the most mature approaches are WSLA [3] and WS-Agreement [2]. WSLA, which was published by IBM in 2003 (no further updates), provides a specification for the definition and monitoring of SLAs in a Web Service environment. The other well established specification was provided by the Grid Resource Allocation Agreement Protocol Working Group of OGF (GRAAP-WG). WS-Agreement defines a language and protocol to represent the services of providers, creating agreements based on offers and also monitors the

agreement compliance at runtime. Both specifications were already taken up by different research communities which tend currently to use a mix of both specifications. The value of using them together seems to be much higher than building only on one or the other individually (cf. TrustCoM [5] and BREIN [6] projects).

With respect to the Negotiation Protocol, a variety of specifications for that area are available. The most referenced one is WS-AgreementNegotiation [7], even though it is still under discussion. So far it follows the Discrete Offer (also known as Take-It-Or-Leave-It) approach which is simple to implement but not efficient and flexible enough for complex application areas. Alternatives like multi-round negotiation or auctions are also examined, but have never reached the status of a well balanced specification for the Web Service domain. Especially auctions [8] are mainly settled in the multiagent domain but rarely examined by Web Service research activities.

Even though there were many thoughts about semantic enhancements of SLAs, not many activities really provided results originating from this research topic. Frankova et al [9] present an approach showing a formal definition of WS-Agreement, based on the concepts of finite automata. Even though the approach is very interesting, it uses semantic concepts in a different way and therefore a comparison with the work shown in this paper is nearly impossible.

Oldham et al [10] present the Semantic WS-Agreement Partner Selection (SWAPS). This approach covers how to overcome limitations of WS-Agreement with respect to syntactical matching; it enables intelligent partner selection based on SLA technologies. These limitations exist due to the scantly defined XML domain vocabulary of WS-Agreement. The main focus of SWAPS is on closing these gaps, by adding more structure to the original specification, to enable automatic parsing and reasoning. This additional structure is based on WSLA and introduces the OntConcept element in the schema, which clarifies the ambiguities of agreements by linking expression parameters directly to the concrete ontology concepts. This paper works further in the direction of SWAPS by using semantic annotation and also extending the idea towards negotiation.

3 The SA-SLA Specification

Customers and service providers need a common language for SLA negotiation, in order to understand each other's offers and bids. Current SLA specifications only define the format of expressing an SLA offer, but the content can use different languages, terms and metrics. To solve the problem, one possibility could be to construct an XML schema definition of SLA metrics, but the general experience is that a semantic definition supports better the re-use, re-combination and translation of descriptive elements.

In this paper we propose the usage of semantic annotations inside current SLA descriptions. A semantic annotation is additional information that identifies or defines a concept in a semantic model in order to describe part of any document element. This way of annotating does not require an extension of the XML Schema, but it uses the extensibility elements of the XML schema as placeholders for the annotations. Thus, annotations are simply ignored by tools that cannot interpret the additional semantics,

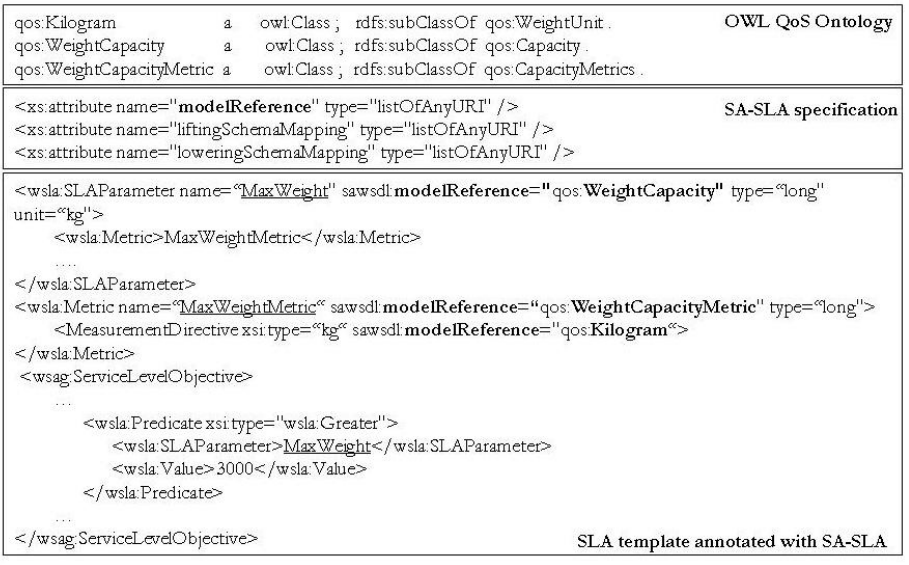

Fig. 2. Samples for the SA-SLA approach

allowing a "mixed economy" environment where annotated and non annotated SLAs can co-exist.

This work proposes a specification for semantic annotations in SLA files called Semantic Annotations for Service Level Agreement (SA-SLA), which is based on the Semantic Annotations for Web Service Description Language (SA-WSDL) [11]. SA-SLA provides a description format extending current WS-Agreement specification [2] with semantic annotations in order to provide dynamically linkable domain vocabularies for the SLA format. Thus, SLA elements can be linked to semantic concepts defined externally to the SLA.

As well as SA-WSDL, this specification is comprised mainly of: i) model reference, which is an association between a SLA schema and a concept in some semantic model, and ii) schema mappings, specifying mappings between semantic data and XML [11].

Figure 2 shows how it is possible to link elements in a SLA template file such as SLA Parameters or metrics (MaxWeight) with concepts belonging to the OWL QoS ontology (WeightCapacity, Kilogram) by using the SA-SLA specification. As a result, we have a SLA template annotated semantically.

Although we use the same elements as SA-WSDL, our approach is different since the annotated elements are different. The former is using WSDL elements while SA-SLA annotates WS-Agreement and WSLA elements. Concretely, these elements are service descriptions (terms related to functional properties), SLA parameters, metrics and service properties, since they constitute the basic vocabulary inside the SLA message structure.

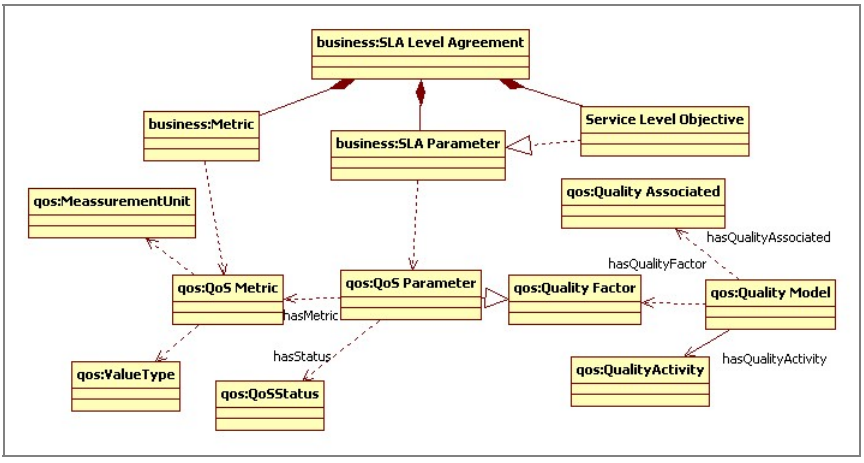

Fig. 3. Key concepts of our conceptual model

4 SA-SLA Related Ontologies

The SA-SLA allows to link SLA elements with concepts from a conceptual model formalized as ontologies. Thus, we also provide a common conceptual model for SLAs, which can be used to annotate SLA offers and bids, and which can be extended in local repositories according to the parties' requirements. The common conceptual model used in SA-SLA is mainly composed of concepts belonging to the core BREIN ontologies [6]; the business ontology and QoS ontology, whose main elements are summarized in Figure 3.

The business ontology enables the representation of SLAs as OWL instances in the sense of WS-Agreement and WSLA specifications. The emphasis is made on the semantic representation of Service Level Objectives (SLOs). The internals of a SLO are left undefined in WS-Agreements. Thus, they are modeled according to WSLA (as under SLA parameter), and include the name and value of the quality attribute, plus the metric used to calculate the value. Moreover, the SLA parameter is related to the QoS Parameter in the QoS ontology.

The QoS ontology, which is also part of the business ontology, collects the metrics and quality attributes to be used in SLOs. The basic concepts are taken from the quality model defined by OASIS in the Web Services Quality Model (WSQM) specification [12]. WSQM complements existing SLA-related specifications with a general view on quality related roles, processes and attributes. WSQM uses the term Quality Factor for QoS parameters and further categorizes it into sub-factors and layers concerning the user's view, interoperability and management. In our ontology, each QoSParameter is associated with a Metric characterized by ValueType (float, integer, boolean, etc.), a Value and a MeasurementUnit (e.g. euro, kB, ms). Finally, the QoSParameter can have several statuses depending on if it is requested by a customer, or offered by a provider, etc.

Involved parties can extend the common ontology to consider their internal requirements forming local ontologies. These local ontologies extend the common ontology

with locally used parameter types and also with local technological knowledge, which enables a better understanding of SLA requests by the provider in terms of its own local infrastructure. Service providers can add here the definition of locally used QoS parameters, metrics or measurement units. They can also add descriptions about local environment, such as available resource types and their parameters, licenses or platform dependencies. Furthermore, the mapping of received SLAs into the local environment can be supported with such local ontologies, either by new instances (such as conversion rates) or by conversion rules. In the next sections we present examples of these rules and demonstrate that the approach works on available Semantic Web technologies and it provides an adaptable solution without changing program code.

5 Application of SA-SLA for Negotiation

Based on WS-Negotiation [4], our negotiation approach follows the three-layered approach (Figure 1), so that it provides a protocol-independent solution on the level of Negotiation Message and a practical framework on the level of Negotiation Decision Making. For the Negotiation Message layer initially a merge of WS-Agreement [2] and WSLA [3] is used, allowing rich SLA definition.

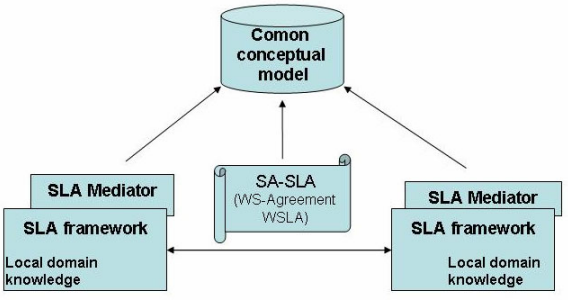

Fig. 4. SLA negotiation framework enhanced to use SA-SLA

The application of the framework for SLA negotiation is depicted on Figure 4. The common part contains the SA-SLA specification defining how to annotate SLA files and the common conceptual model expressed as an ontology. The model consists of a generic quality model and the structure of SLA documents as well as metrics and measurement units for SLA parameters.

The parties involved in SLA negotiation usually have their existing frameworks supporting SLA management. The negotiation may take place between customer and provider and also between two providers (outsourcing scenario), by exchanging SA-SLA files. The role of the SLA Mediator component is to import received SLAs into the local SLA model, which is used by the SLA framework. To achieve this, the SLA Mediator combines the common model with local knowledge.

This approach allows to keep the original standard SLA formats, and to explain its elements semantically through semantic annotations in SLA template files. The

ontologies used for annotations provide the common language of SLA negotiations, while the XML format files provide the common format or syntax. By this current approach, interoperability between entities involved in SLA negotiation is improved in a backward compatible way, so that components non aware of semantic annotations can continue working with SLA files. Furthermore, it is a lightweight and scalable approach, where participants can choose between various levels of adaptation to SA-SLAs.

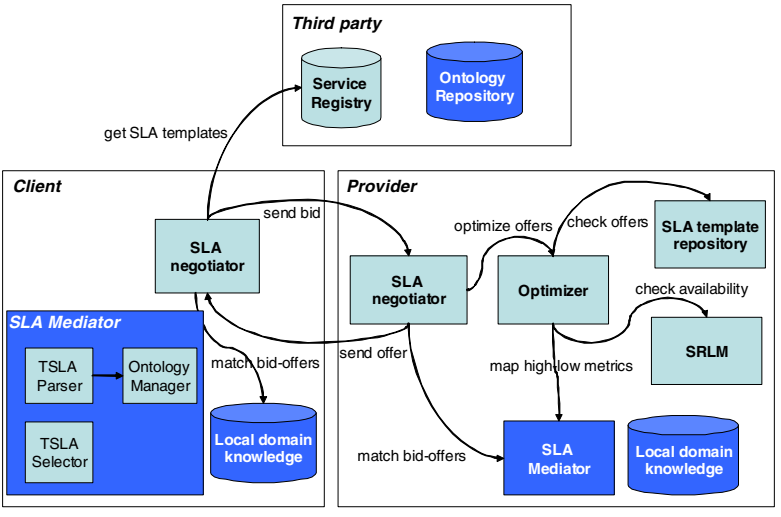

Fig. 5. Concrete implementation of the SLA negotiation framework

In the following we present the concrete implementation of the previously described generic approach on the basis of an existing negotiation framework, which is represented in Figure 5 with lighter shade. The framework is composed of the following components; the SLA Negotiator in the customer side which acts on behalf of the customer to achieve the agreement with the provider. It has to both create bids and evaluate the offers received from providers. The SLA Negotiatior in the Provider side processes incoming SLA requests from customers. The SLA Negotiator has been implemented with the Globus Toolkit 4 middleware. It follows the principles and designs of the WS-Agreement and provides the means of optional agent usage for negotiation. With this the possibility of going away from the traditional one-phase commit protocol to multiphase negotiation (e.g. with FIPA Iterated Contract Net Protocol [13]) is given. Moreover, it optimizes the offers sent using the Optimizer component, which is able to map the high level terms of the SLA Offer to the low level infrastructure level (checking the resources availability with the Semantic Resource Lifecycle Manager, SRLM) and available templates in the repository).

The base framework is extended with new components represented in Figure 5 by the darker shade. A third party hosts the common conceptual model in the Ontology Repository, which stores the ontologies used to annotate the SLA templates files. With that it provides the semantic model in charge of improving the interoperability.

Each of the entities involved in negotiation has a Local Domain Knowledge, which includes all business and negotiation rules and enterprise knowledge involved in the negotiation. Finally, the SLA Mediator matches SA-SLA requests and offers, by importing SLAs into the local SLA model supported and used within the framework.

The SLA Mediator allows for matching customer SLA requests with providers' offers. It is implemented in Java, using Jena as the semantic platform. First, the TSLA Parser parses the incoming SLA files performing two kinds of matchmaking depending on whether the metrics are annotated or not: i) syntactically, which involves a keyword search and ii) semantically, which interacts with the ontology manager in order to match the semantic QoS metrics with the customer' requests.

The Parser creates the OWL representation of the incoming SLA by extending the local domain ontology with new facts through the Jena OntModel API [15].

The Ontology Manager (Jena) allows the management of the local domain ontology. Pellet [16] is run inside Jena to provide OWL DL inferencing and SWRL support. The domain rules are formalized using SWRL [17]. The TSLA Selector provides a comparison (ranking) of local SLA templates (offers) and the incoming customer bid. This ranking is a number between 0 and 1 representing how well the template fits the request. This outcome is then used by the SLA Negotiator to create the offer and send it as a response to the customer.

6 Experimentation

The scenario chosen to demonstrate the applicability of the approach is obtained from the Airport Scenario of the BREIN project [6]. This scenario tries to demonstrate the coordination mechanism for ad-hoc resource allocation.

Different resource providers (ground-handling companies) are involved in the collaboration value chain of the airport ground handling. Among the different providers and customers an agreed SLA is required. Figure 6 shows examples for the parameters and metrics the customers and providers may use. It is apparent that the parameters are related, but they are not uniformly expressed, which demonstrates the heterogeneity problems in terms of:

- Different **service attribute and value abstractions:** e.g. "capacity" (abstract) vs. "max. number of passengers" (concrete), "jet-propelled aircraft" (abstract) vs. "B747-400" (concrete)
- Different **service attribute definitions:** e.g. "price" without or with VAT or "price" based on distance from gate to position vs. flat distance,
- Different **service attribute names:** e.g. "max. passengers", "passengers, max.**",** "passengers no.", and the problem of multi-linguality in names,
- Different **units of measurement (UOM) for service attributes:** e.g. kilograms, tons, pounds, park position code vs. GPS coordinate.

In order to improve interoperability between all involved parties, providers and customers define the SLA template files relating their parameters and metrics to a common conceptual model by following SA-SLA (cf. Figure 2). Both the customer's requests and providers' templates are formalized as WS-Agreement (plus WSLA elements) extended by the SA-SLA specification. Coming back to the example

provided in Figure 6, there are two providers with associated templates, which express the metrics and parameters commented in the example by using the SA-SLA specification. On the other side, the customer expresses the SLA customer request by using the same format. In order to formalize the local domain knowledge, the provider creates a set of rules by using concepts from the common ontology.

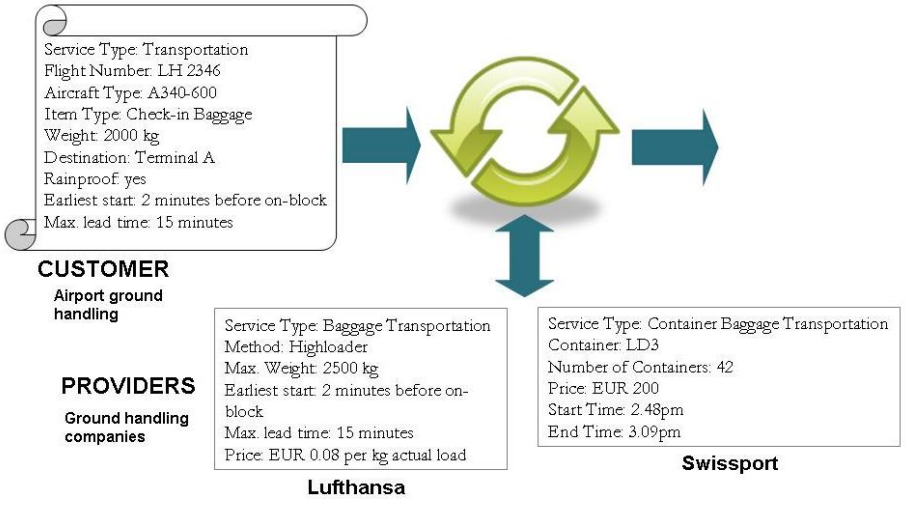

Fig. 6. Examples for SLA bid and offers

These SA-SLA files are then applied in the SLA negotiation, where the semantic annotations in SA-SLA files are used, so that the provider can understand the customer requirements and search for a relevant offer according to the customer bid. Steps required in the process include the following:

Create SLA instances: All the SLA templates in the provider side are stored as OWL instances in the local domain knowledge repository.

Receive bid: when receiving a bid the (provider) SLA Negotiator evaluates if it can provide an offer to satisfy the bid. For this, it calls the SLA Mediator.

Bid conversion: The customer SLA request is parsed by the SLA Mediator and then converted into an OWL representation. An OWL instance represents the SLA request, containing instances for each SLO.

SLO translation: The SLOs of the bid are translated into the corresponding local SLOs using local metrics and units of measurement. In this phase SWRL rules are used to fill 'localized' instance properties. An example given here is a generic conversion rule where conversion 'rates' are defined in OWL:

qos:WeightCapacityMetric(?param) ∧ qos:hasUnit(?param, t) ∧
qos:parameterValue(?param, ?v1) ∧ swrlb:multiply(?v2,?v1,1000) →
hasLocalUnit(?param, kg) ∧ hasLocalValue(?param, ?v2)

Naturally, the provider can define additional, custom rules to provide more complex metric conversions.

Matching phase: Available offers need to be related to the request in this step. First, a new Match instance is created for each offer template using Java code. Then, comparable pairs of SLOs in request and potential offers (from templates) are detected using an SWRL rule:

```
qos:QualityModel(?b) ∧ qos:hasQualityFactor(?b, ?p1) ∧ qos:hasMetric(?p1, ?m1) ∧
hasLocalMUnit(?p1, ?u1) ∧ QualityModelTemplate(?t) ∧ qos:hasQualityFactor(?t, ?p2) ∧
qos:hasMetric(?p2, ?m1) ∧ qos:hasMUnit(?p2, ?u1) ∧ differentFrom(?p1, ?p2)
 → isMatchedTo(?p1, ?p2)
```

The pairs identified by "isMatchedTo" are evaluated one bye one, and accepted if the SLO of the offer is stronger than the SLO of the request, with rules like this one:

```
isMatchedTo(?p1, ?p2) ∧ hasLocalValue(?p1, ?v1) ∧ qos:hasOperation(?p1,
qos:greaterequal) ∧ qos:parameterValue(?p2, ?v2) ∧ qos:hasOperation(?p2, qos:greaterequal)
∧ swrlb:greaterThanOrEqual(?v2, ?v1)
→ isFitting(?p1, ?p2)
```

This is again a place for local adaptation by rules; for example the provider may define a 5% threshold of SLO comparison instead of strict <= or >= comparisons.

Template Scoring: The previous evaluation is used to assign a numeric, normalized score to each SLA template, which represents the fitness of the template for the customer request.

Respond with offer: Finally, the SLA Negotiator collects the ranked list of templates, takes the one with the best score, and creates an offer from it, which is sent to the customer.

7 Conclusions

The paper presents a novel approach towards the improvement of SLA negotiation, which was driven from the need to improve the flexibility of existing SLA negotiation protocols by infusing semantic annotations in a non-intrusive manner. The annotation technique adopted in the proposed SA-SLA specification extends the idea introduced in SAWSDL. Our experimentation with SA-SLA and testing with existing SLA negotiation components proved its backwards compatibility which was the first objective of our approach. The second objective of our experimentation was to demonstrate how semantic interoperability issues found in the annotated WS-Agreement documents were automatically dealt with by the SLA mediator. Appropriate improvements to the SLA negotiation component were made in order to allow the use of the semantic annotations while new ontologies and rules were developed to capture the conceptual model related to the negotiation of logistics services. The conceptual model for our experimentation was derived from the analysis of real requirements from ground-handling services in an airport. The modified SLA negotiation component successfully used semantic technologies to overcome the semantic interoperability issues in an automated fashion.

Acknowledgment

This work has been supported by the BREIN project (http://www.gridsforbusiness.eu) and has been partly funded by the European Commission under contract FP6-034556.

References

1. Mitchell, B., Mckee, P.: SLAs a Key Commercial Tool Innovation and the Knowledge Economy: Issues, Applications. Case Studies (2005)
2. GRAAP-WG: Web Services Agreement Specification (WS-Agreement) (March 14, 2007), http://www.ogf.org/documents/GFD.107.pdf
3. Ludwig, H., Keller, A., Dan, A., King, R.P., Franck, R.: Web service level agreement (WSLA) language specification, http://www.research.ibm.com/wsla/WSLASpecV1-20030128.pdf
4. Hung, P., Li, H., Jeng, J.-J.: WS-Negotiation: An Overview of Research Issues. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences, HICSS 2004 (2004)
5. TrustCom consortium: The TrustCom Project, http://www.eu-trustcom.com/
6. BREIN consortium: The BREIN Project, http://www.eu-brein.com/
7. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Pruyne, J.: Web Services Agreement Negotiation Specification (WS-AgreementNegotiation) (2004)
8. Sandholm, T.W.: Distributed rational decision making. Multiagent systems: a modern approach to distributed artificial intelligence, pp. 201–258. MIT Press, Cambridge (1999)
9. Frankova, G., Malfatti, D., Aiello, M.: Semantics and Extensions of WS-Agreement. Journal of Software 1(1) (July 2006)
10. Oldham, N., Verma, K., Sheth, A., Hakimpour, F.: Semantic WS-agreement partner selection. In: Proceedings of the 15th international conference on World Wide Web, pp. 697–706. ACM, New York (2006)
11. Kopecky, J., Vitvar, T., Bournez, C., Farrell, J.: SAWSDL: Semantic Annotations for WSDL and XML Schema. IEEE Internet Computing 11(6), 60–67 (2007)
12. Kim, E.: OASIS Quality Model for Web Services. Version 2.0 (2005)
13. FIPA Iterated Contract Net Interaction Protocol Specification, http://www.fipa.org/specs/fipa00030/SC00030H.html
14. Munoz, H., Kotsiopoulos, I., Vaquero, L.M., Rodero, L.: Enhancing Service Selection by Semantic QoS. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 565–577. Springer, Heidelberg (2009)
15. Carroll, J.J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K.: Jena: implementing the semantic web recommendations, New York, NY, USA, pp. 74–83 (2004)
16. Evren, S., Bijan, P., Bernardo Cuenca, G., Aditya, K., Yarden, K.: Pellet: A practical OWL-DL reasoner. Web Semant. 5(2), 51–53 (2007)
17. O'connor, M., Knublauch, H., Tu, S., Grosof, B., Dean, N., Grosso, W., Musen, M.: Supporting rule system interoperability on the semantic web with SWRL, pp. 974–986 (2005)