On Using Semantic Web Query Languages for Semantic Web Services Provisioning

José María García, Carlos Rivero, David Ruiz and Antonio Ruiz-Cortés
Dept. Lenguajes y Sistemas Informáticos, University of Seville
ETS Ingeniería Informática, Av. Reina Mercedes s/n, 41012 Sevilla, Spain

Abstract—*Although there are several approaches to discover Semantic Web Services based on Description Logics reasoning, the use of standard Semantic Web query languages for this task is not so widely spread, partly because service discovery involves some issues that these languages do not usually deal with, such as complex matching, results ranking or interoperability. In this work we analyze the suitability of existing query languages to perform provisioning tasks (namely discovery, ranking and selection) within a Semantic Web Services scenario. Additionally, the requirements a Semantic Web query language has to fulfill in order to be used within a provisioning scenario are enumerated, giving some insights into how to extend current query languages to do so. Furthermore, an analysis of current provisioning proposals achievement of those requirements is presented.*

Keywords: Service Discovery, Semantic Web Services, Ontology Languages, Query Languages, Semantic Web.

1. Introduction

Once a service has been published and made available from a repository, potential users can fetch for desired services. This fetching, referenced as service provisioning or procurement [1], involves different sequential tasks, namely discovery, ranking and selection. Firstly, services that fulfill the user requirements are discovered. Secondly, those services are ranked with respect to user preferences. Finally, the best ranked service is selected so it can be executed later on.

Usually, semantic discovery is considered as a functional filter, because at this stage the user is looking for a service that provides a requested functionality. Current discovery approaches present matchmaking algorithms that are highly coupled with the service representation formalism, often based on Descriptions Logics [2]–[7]. These proposals defines matching degrees that measure the similarity between the user requirements and the available service descriptions. Furthermore, current proposals rely on service descriptions and user preferences defined using OWL-S [8] and WSMO [9] ontologies.

Ranking and selection often involves non-functional properties defined over services, e.g. cost or availability. These properties are used to obtain a ranking of discovered services, so the best service, in terms of user preferences which are based on said properties, can be selected. Current proposals provide ontologies to express non-functional properties about services that are used within ad-hoc ranking algorithms [10]–[13]. As with discovery approaches, ranking and selection proposals have a high coupling between preference description formalisms and algorithms used to perform these tasks.

The question that arise in this provisioning scenario is: why current proposals are not using a Semantic Web query language to perform discovery, ranking and selection? One reason can be found at the level of maturity of these query languages so that, until recently there has not been a standard query language for the Semantic Web. Nevertheless, the sometimes complex reasoning needed to match services and user preferences conforms a key feature that current query languages do not completely support, especially RDF-based languages. In this paper we depict the requirements a query language has to fulfill in order to be used to discover and rank Semantic Web Services (SWS), thoroughly analyzing the suitability of current query languages for provisioning tasks, and discussing extensions that make these languages compliant with the enumerated requirements, at least in part.

The rest of the paper is structured as follows. In Sec. 2 existing query languages for the Semantic Web are described. Then, in Sec. 3 an analysis of the requirements for these query languages to support SWS provisioning is presented, along with a discussion about current proposals on the topic. Finally, in Sec. 4 we sum up our contributions, and discuss our conclusions.

2. Query Languages for the Semantic Web

One of the most important features of the Semantic Web is that it separates the data information from the schema model to be applied to this information [14]. In the Semantic Web field, the W3C recommends RDF (Resource Description Framework) [15] as the data model, and RDFS (RDF Schema) [16] or OWL (Web Ontology Language) [17] as the schema model.

OWL is divided in three increasingly expressive sublanguages: Lite, DL and Full. DL stands for Description Logics, which is a logical formalism that provides the theoretical foundations for Semantic Web ontologies [18],

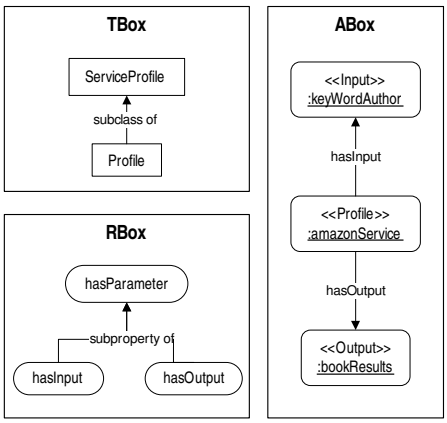

Fig. 1: DL ontology components

[19]. A DL ontology has three conceptual components: assertions about classes (TBox), assertions about properties and property hierarchies (RBox), and property assertions between individuals and membership assertions (ABox). An example of this three conceptual components is presented in Fig. 1, where some parts of the profile ontology of OWL-S specification [8] are shown. Thus, in the TBox, there are two classes, *Profile* and *ServiceProvile* and a *subclass of* assertion between them. In the RBox, two *subproperty of* assertions are shown, between *hasInput*, *hasOutput* and *hasParameter* properties. Then, in the ABox example, there are three membership assertions (*keyWordAuthor typeOf Input*, *amazonService typeOf Profile*, and *bookResults typeOf Output*, represented by the word in double angle brackets), and two property assertions (*hasInput(amazonService, keyWordAuthor)* and *hasOutput(amazonService, bookResults)*).

There exists two main approaches in Semantic Web query languages: RDF-based and DL-based query languages [20], [21]. On the one hand, RDF-based query languages allow to fetch RDF triples based on matching triple patterns with RDF graphs. On the other hand, DL-based query languages allow to query OWL-DL ontologies with TBox queries, RBox queries, ABox queries or any combination of them.

Concerning RDF-based query languages, there are several approaches with different features. In [20], Bailet *et al.* survey 26 different query languages such as SeRQL [22], RQL [23] or RDQL [24]. At present, the great majority of these languages have neither any implementation nor an updated implementation. This is caused by the fact that SPARQL is the only language that is a W3C recommendation [25]. In fact, SPARQL is fully supported in several implementations[1]. Other surveys of pre-SPARQL languages can be found at [22], [26].

SPARQL has four different types of queries: `SELECT`, `CONSTRUCT`, `DESCRIBE` and `ASK`. Each type serves for a different purpose: `SELECT` queries return variables and their bindings directly; `CONSTRUCT` queries build an RDF graph based on a template defined in the query; `ASK` queries test whether or not a pattern has any solution; and `DESCRIBE` queries return an RDF graph not based on a template in the query (as in `CONSTRUCT` queries) but on a pre-configured graph.

The SPARQL Working Group has already detected some extensions to be applied to the current specification[2]. Some of these extensions are: insert/update/delete queries, access to collection members, or aggregate functions (`COUNT`, `SUM`, `GROUP BY`, etc).

Besides these extensions, others have already been proposed in the Semantic Web research field. For instance, Kiefer *et al.* present iSPARQL [27] which supports customized similarity functions to query RDF graphs. These functions are used in different Semantic Web research fields such as semantic data integration or ontology matching. PSPARQL [28] extends the original recommendation allowing regular expressions in the predicates of the graph patterns, while CPSPARQL [29] is an extension of PSPARQL that introduces constraints on paths. Moreover, SPARQL2L [30] and SPARQLeR [31] are very related to PSPARQL and support the discovery of semantic associations which are undirected paths connecting two entities of an ontology.

The SPARQL standard is designed to query RDF data only, not including RDFS vocabulary. Although RDF is a data format representing a directed labeled graph, SPARQL only provides limited navigational functionalities. nSPARQL [32] is another extension of SPARQL which allows to query RDF data according to the semantics of RDFS. This extension uses recursive graph paths to achieve its goal. Finally, Siberski *et al.* [33] present an extension which supports the expression of preferences and ranking that is further discussed in Sec. 3.2.

Regarding DL-based query languages, SPARQL-DL [21] is aligned with SPARQL to improve the interoperability of applications on the Semantic Web, and can be implemented on top of existing OWL-DL reasoners because of its simplicity. A preliminary prototype of SPARQL-DL has been implemented on top of the OWL-DL Pellet reasoner [34].

OWL-QL [35] is a language and protocol for query/answer dialogues in which Semantic Web computational agents are involved. These agents use OWL ontologies to make the dialogues possible. OWL-QL is designed to be easily adaptable to other declarative formal logic representations such as RDF or RDFS. Other DL-based query languages are OWL SAIQL [36] or SWQL [37].

Although DL-based query languages provide more reasoning mechanisms than RDF-based ones, the former are not mature enough and they are in early stages of development [20]. In this field, SPARQL-DL is the most promising one

[1]http://www.w3.org/2001/sw/DataAccess/tests/implementations

[2]http://www.w3.org/2009/01/sparql-charter

because, as stated before, it is included in the well-known Pellet reasoner, which is in continuous development.

From the presented survey we conclude that, though there are several Semantic Web query languages, SPARQL is the most widely used, partly because it is a W3C recommendation. In fact, it seems to be the chosen query language to be applied to SWS provisioning processes. However, its lack of reasoning mechanisms *per se* makes necessary to extend it in order to allow flexible matchmaking of services. The needed extensions to support this provisioning scenario are introduced in the next section.

3. SWS Provisioning Using Query Languages

As introduced in Sec. 1, SWS are usually discovered in terms of a requested functionality. This process basically applies a functional filter to a service repository, so a set of compliant services are returned to the user. Although service discovery could make use of Semantic Web query languages to define the filter applied, only a few proposals actually use them [33], [38], [39]. Additionally, once services have been discovered, they have to be ranked so the user can select the best one in term of stated preferences. These tasks could be also performed using a query language, separately from discovery [40], or in the same query [38].

In the following, we analyze what are the requirements a Semantic Web query language has to satisfy in order to support SWS provisioning tasks, and then discuss to what extent current proposals that use query languages to perform these tasks fulfill the identified requirements.

3.1 Requirements Analysis

Considering the SWS provisioning scenario, where a user wants to fetch the best service from a repository in terms of his or her preferences, the query language chosen to support it has to be able to describe queries for service discovery and ranking. In order to do so, we have identified seven requirements that are enumerated in the following:

(R1) *Based on standards*. Currently, SPARQL is the proposed standard query language for the Semantic Web, so any provisioning approach that wants to use query languages in its process should be based on SPARQL, possibly extending it or using one of its currently published extensions.

(R2) *Compatible with SWS frameworks*. There are three main frameworks and ontologies to define SWS: OWL-S [8], WSMO [9], and SAWSDL [41]. Any query language used in a provisioning scenario has to be capable to handle SWS descriptions from any of the enumerated frameworks.

(R3) *Support for complex matching and similarity degrees*. Services are not always described using exactly the same domain as user requests, so there is a need for some reasoning about equivalences and similarity degrees between concepts, especially useful in discovery where soft matching is needed [7]. For instance, iSPARQL supports this kind of flexible matching [27].

(R4) *Reasoning mechanisms*. Related to the previous requirement, reasoning is a key feature to support interoperability and soft matching between concepts being used in queries. DL-based query languages offer some facilities to fulfill this requirement, such as SPARQL-DL which is evaluated by Pellet reasoner. Some proposals perform the reasoning before the query execution, updating the knowledge base and then executing the query.

(R5) *Evaluation mechanisms*. Especially in the ranking process, evaluation mechanisms are needed in order to compute preference values used to rank discovered services. Again, DL-based query languages offer limited support for this requirement, but some preferences can be computed easier using different evaluation mechanisms within a hybrid approach [40], especially when continuous domains are involved.

(R6) *Facilities to order results by computed values*. After discovering, services have to be ranked in terms of user preferences, so a query language should provide facilities not only to evaluate those preferences, but to order the resulting values using different ordering policies. Standard SPARQL offers a basic ordering support in its `ORDER BY` clause.

(R7) *Decoupled from formalism*. Queries have to be generic and not coupled with the actual techniques used to evaluate them. Thus, different implementations of the reasoning and evaluation mechanisms can be used and changed dynamically, depending on the expressiveness of user preferences.

The discussed requirements list make a convenient framework to compare different proposals which use query languages to perform provisioning task. This list is used in the next section for that purpose.

3.2 Discussion of Current Proposals

There are some proposals that use a Semantic Web query language to perform discovery, ranking and selection of services. They choose SPARQL as their base language, though some extensions have to be added to fully support provisioning tasks, i.e. to fulfill some of the requirements we have identified before.

Thus, Lamparter *et al.* [38] provide an ontology to represent service offers and requests that conforms the foundations for a discovery and selection process performed using rules in SWRL [42] and SPARQL queries. These queries includes predicates that have to be evaluated at run-time, so they include an extension to SPARQL that is implemented using different proposed algorithms. Thus, a generic query

Table 1: Requirements satisfied by discussed proposals

	Lamparter *et al.* [38]	Iqbal *et al.* [39]	Siberski *et al.* [33]
R1	$\surd$	$\surd$	$\surd$
R2	$\surd$	$\surd$	$\sim$
R3	$\sim$	$\times$	$\times$
R4	$\sim$	$\times$	$\times$
R5	$\surd$	$\times$	$\sim$
R6	$\surd$	$\times$	$\surd$
R7	$\times$	$\times$	$\times$

for a user request is provided, though this query depends on rules that change the matchmaking policy, e.g. allowing matching degrees as in [7].

Another discovery approach that uses SPARQL to actually perform semantic service discovery is proposed by Iqbal *et al.* in [39]. In this case, the authors embed semantic information about services using SAWSDL, which is an extension to add semantics to WSDL descriptions [41]. Thus, they define pre and post-conditions of services using SPARQL `CONSTRUCT` queries so that depending on each service functionality, they add corresponding RDF tuples representing that functionality to the knowledge base. Then, their discovery algorithm use an `ASK` query to check whether a service fulfills a user request or not, returning the results.

Finally, concerning ranking, there is another approach presented in [33], where Siberski *et al.* propose an extension to SPARQL so that preferences are described directly using the query language, without basing on existing preferences and non-functional properties ontologies, as in other semantic ranking approaches [12], [43]. They provide a `PREFERRING` clause that states preferences among values of variables, similar to `FILTER` expressions. However, this approach does not have the flexibility and reasoning facilities that provides a solution based on an external ontology.

Table 1 shows how well previously discussed proposals match the requirements enumerated in Sec. 3.1. In this table, $\surd$ means a full support of the requirement; $\sim$ indicates that the proposal provides a partial or incomplete materialization of the corresponding requirement; and $\times$ is used when the requirement is not sufficiently supported.

From this comparison, several conclusions can be obtained. Firstly, the most complete proposal is the one presented by Lamparter *et al.* [38]. Its main drawback is that its matching (R3) and reasoning mechanisms (R4) depends on logic rules that the user must provide. In addition, though it is able to rank services in terms of complex preferences that are evaluated at run-time, the formalism and algorithms used for that evaluation are explicitly expressed using rules, causing a not desired coupling (R7).In the case of Iqbal *et al.* [39], they use standard SPARQL without extensions in a SAWSDL description, so only the first and second requirements are met. Finally, Siberski *et al.* [33] offer an interesting approach to fulfill requirement R6 by extending SPARQL both syntactically and semantically, but the rest of the requirement are not completely supported.

In general, we conclude that the main limitations of current approaches are, on the one hand, their lack of mechanisms to perform complex matchings and reasoning tasks (requirements R3, R4, and to a lesser extent R5), and on the other hand, their high coupling between description formalisms and algorithms used to evaluate the queries (R7).

4. Conclusions

Semantic Web query languages have not been used for SWS provisioning until recently. However, some proposals are emerging in the field, which are mainly based on SPARQL. There are also several extensions to SPARQL that can be adopted by SWS provisioning proposals which have been discussed thoroughly in this paper. Furthermore, we have provided a list of requirements that query languages and their extensions have to meet in order to be useful within a provisioning scenario. This requirements analysis also provides a convenient framework to compare current and ongoing researches on query languages to discover and rank SWS.

Additionally, in this work we have discussed some proposals, concluding that they partly fulfill those requirements, but there are some areas that need further research. In particular, matching, reasoning and evaluation mechanisms have to be worked out, but taking care of the level of coupling these mechanisms have with respect to definition formalisms.

Acknowledgment

This work has been partially supported by the European Commission (FEDER) and Spanish Government under CICYT project Web-Factories (TIN2006-00472) and by the Andalusian Government under project ISABEL (TIC-2533).

References

[1] A. Ruiz-Cortés, O. Martín-Díaz, A. Durán-Toro, and M. Toro, "Improving the automatic procurement of web services using constraint programming," *Int. J. Cooperative Inf. Syst*, vol. 14, no. 4, pp. 439–468, 2005.

[2] J. González-Castillo, D. Trastour, and C. Bartolini, "Description logics for matchmaking of services," Hewlett Packard Labs, Tech. Rep. HPL-2001-265, 2001.

[3] L. Li and I. Horrocks, "A software framework for matchmaking based on semantic web technology," in *Int. World Wide Web Conference*, 2003, pp. 331–339.

[4] C. Lutz and U. Sattler, "A proposal for describing services with DLs," in *Int. Workshop on Description Logics*, 2002.

[5] E. Motta, J. Domingue, L. Cabral, and M. Gaspari, "IRS-II: A framework and infrastructure for semantic web services," in *Int. Semantic Web Conference*, 2003, pp. 306–318.

[6] N. Srinivasan, M. Paolucci, and K. Sycara, "Semantic web service discovery in the OWL-S IDE." in *Hawaii International Conference on Systems Science*, 2006.

[7] K. Sycara, M. Paolucci, A. Ankolekar, and N. Srinivasan, "Automated discovery, interaction and composition of semantic web services." *J. Web Sem.*, vol. 1, no. 1, pp. 27–46, 2003.

[8] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. Mcdermott, *et al.*, "OWL-S: Semantic markup for web services," DAML, Tech. Rep. 1.1, 2004.

[9] D. Roman, H. Lausen, and U. Keller, "Web service modeling ontology (WSMO)," WSMO, Tech. Rep. D2 v1.3 Final Draft, 2006.

[10] J. Pathak, N. Koul, D. Caragea, and V. G. Honavar, "A framework for semantic web services discovery," in *WIDM '05: Proceedings of the 7th annual ACM international workshop on Web information and data management*. New York, NY, USA: ACM Press, 2005, pp. 45–50.

[11] E. M. Maximilien and M. P. Singh, "A framework and ontology for dynamic web services selection," *Internet Computing, IEEE*, vol. 8, no. 5, pp. 84–93, 2004.

[12] X. Wang, T. Vitvar, M. Kerrigan, and I. Toma, "A QoS-aware selection model for semantic web services." in *ICSOC 2006*, ser. LNCS, A. Dan and W. Lamersdorf, Eds., vol. 4294. Springer, 2006, pp. 390–401.

[13] C. Zhou, L. Chia, and B. Lee, "DAML-QoS ontology for web services," in *IEEE International Conference on Web Services*, 2004, pp. 472–479.

[14] G. Antoniou and F. vanHarmelen, *A Semantic Web Primer*, 2nd ed. Cambridge, MA, USA: MIT Press, 2008.

[15] D. Beckett, "RDF/XML Syntax Specification," W3C, Tech. Rep., 2004. [Online]. Available: http://www.w3.org/TR/rdf-syntax-grammar/

[16] D. Brickley and R. Guha, "RDF Vocabulary Description Language 1.0: RDF Schema," W3C, Tech. Rep., 2004. [Online]. Available: http://www.w3.org/TR/rdf-schema/

[17] D. L. McGuinness and F. van Harmelen, "OWL Web Ontology Language," W3C, Tech. Rep., 2004. [Online]. Available: http://www.w3.org/TR/owl-features/

[18] A. Fokoue, A. Kershenbaum, L. Ma, E. Schonberg, and K. Srinivas, "The Summary Abox: Cutting Ontologies Down to Size," in *International Semantic Web Conference*, 2006, pp. 343–356.

[19] G. D. Giacomo and M. Lenzerini, "TBox and ABox Reasoning in Expressive Description Logics," in *Description Logics*, 1996, pp. 37–48.

[20] J. Bailey, F. Bry, T. Furche, and S. Schaffert, "Web and Semantic Web Query Languages: A Survey," in *Reasoning Web*, 2005, pp. 35–133.

[21] E. Sirin and B. Parsia, "SPARQL-DL: SPARQL Query for OWL-DL," in *OWLED*, 2007.

[22] P. Haase, J. Broekstra, A. Eberhart, and R. Volz, "A Comparison of RDF Query Languages," in *International Semantic Web Conference*, 2004, pp. 502–517.

[23] G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis, and M. Scholl, "RQL: a declarative query language for RDF," in *WWW*, 2002, pp. 592–603.

[24] A. Seaborne, "RDQL - A Query Language for RDF," HP Labs Bristol, Tech. Rep., 2004. [Online]. Available: http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/

[25] E. Prud'hommeaux and A. Seaborne, "SPARQL Query Language for RDF," W3C, Tech. Rep., 2006. [Online]. Available: http://www.w3.org/TR/rdf-sparql-query/

[26] R. Angles and C. Gutiérrez, "Querying RDF Data from a Graph Database Perspective," in *ESWC*, 2005, pp. 346–360.

[27] C. Kiefer, A. Bernstein, and M. Stocker, "The Fundamentals of iSPARQL: A Virtual Triple Approach for Similarity-Based Semantic Web Tasks," in *ISWC/ASWC*, 2007, pp. 295–309.

[28] J.-F. Baget, F. Alkhateeb, and J. Euzenat, "RDF with regular expressions," INRIA, Tech. Rep., 2007. [Online]. Available: http://hal.inria.fr/docs/00/14/85/17/PDF/RR-6191.pdf

[29] F. Alkhateeb, J.-F. Baget, and J. Euzenat, "Constrained Regular Expressions in SPARQL," in *The 2008 International Conference on Semantic Web and Web Services (SWWS)*. Las Vegas, NV: CSREA Press, Jul 2008, pp. 91–99.

[30] K. Anyanwu, A. Maduko, and A. P. Sheth, "SPARQ2L: towards support for subgraph extraction queries in rdf databases," in *WWW*, 2007, pp. 797–806.

[31] K. Kochut and M. Janik, "SPARQLeR: Extended Sparql for Semantic Association Discovery," in *ESWC*, 2007, pp. 145–159.

[32] J. Pérez, M. Arenas, and C. Gutierrez, "nSPARQL: A Navigational Language for RDF," in *International Semantic Web Conference*, 2008, pp. 66–81.

[33] W. Siberski, J. Z. Pan, and U. Thaden, "Querying the Semantic Web with Preferences," in *International Semantic Web Conference*, 2006, pp. 612–624.

[34] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical OWL-DL reasoner," *J. Web Sem.*, vol. 5, no. 2, pp. 51–53, 2007.

[35] R. Fikes, P. J. Hayes, and I. Horrocks, "OWL-QL - a language for deductive query answering on the Semantic Web," *J. Web Sem.*, vol. 2, no. 1, pp. 19–29, 2004.

[36] A. Kubias, S. Schenk, S. Staab, and J. Z. Pan, "OWL SAIQL - An OWL DL Query Language for Ontology Extraction," in *OWLED*, 2007.

[37] P. Lehti and P. Fankhauser, "SWQL - A Query Language for Data Integration Based on OWL," in *OTM Workshops*, 2005, pp. 926–935.

[38] S. Lamparter, A. Ankolekar, R. Studer, and S. Grimm, "Preference-based selection of highly configurable web services," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*. New York, NY, USA: ACM, 2007, pp. 1013–1022.

[39] K. Iqbal, M. L. Sbodio, V. Peristeras, and G. Giuliani, "Semantic service discovery using SAWSDL and SPARQL," in *Semantics, Knowledge and Grid, 2008. SKG '08. Fourth International Conference on*, 2008, pp. 205–212.

[40] J. M. García, D. Ruiz, and A. Ruiz-Cortés, "Semantic discovery and selection: A qos-aware, hybrid model," in *The 2008 International Conference on Semantic Web and Web Services (SWWS)*. Las Vegas, NV: CSREA Press, Jul 2008, pp. 3–9.

[41] J. Farrell and H. Lausen, "Semantic annotations for WSDL and XML Schema," W3C Recommendation, World Wide Web Consortium, Tech. Rep., August 2007. [Online]. Available: http://www.w3.org/TR/sawsdl/

[42] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A semantic web rule language combining OWL and RuleML," W3C Member Submission, Tech. Rep., 2004.

[43] J. M. García, I. Toma, D. Ruiz, and A. Ruiz-Cortés, "A service ranker based on logic rules evaluation and constraint programming," in *2nd ECOWS Non-Functional Properties and Service Level Agreements in Service Oriented Computing Workshop*, ser. CEUR Workshop Proceedings, vol. 411, Dublin, Ireland, Nov 2008.