A Semantic Scoring Approach for Service Offers

Ikbel Guidara, Kaouthar Fakhfakh, and Tarak Chaari

ReDCAD Laboratory, University of Sfax
B.P. 1173, 3038 Sfax, Tunisia
{ikbel.guidara,kouthar.fakhfakh}@gmail.com,
tarak.chaari@redcad.org

Abstract. Automating service selection using semantic approaches have been extensively studied in recent years. In fact, given the big number of provider offers, sourcing of the most relevant service to the client intentions is a complex task especially when providers and customers don't share the same knowledge degree. In particular, differentiating between very similar offers satisfying the same number of client constraints is still a challenging task. In this paper, we present a novel semantic scoring approach that helps clients to select the most appropriate service offer according to their intentions. Our approach detects direct and indirect semantic correspondences between these intentions and the available offers using ontological models. It fairly evaluates these offers and ranks them according to their semantic closeness to the client intentions taking into account both functional and QoS properties. Our ranking is based on a deep examination of provider offers and can distinguish between services that look the same for non expert clients.

Keywords: Ontologies, Quality of Service (QoS), Semantic Web, Service Sourcing.

1 Introduction

With the continued growth of Service Oriented Architectures, the sourcing of the most relevant service becomes a challenge. In fact, generally, providers define their services using fixed and predefined choices and technical terms [5]. Because of their limited knowledge, clients may not understand these complex terms. As a result, they may not choose the best offer and may select a service that does not adequately satisfy their needs. In addition, there is often the case when several services fulfill client's requirements and satisfy the same number of client's constraints. Among these services, which one will be selected is a very difficult task. This issue is still insufficiently tackled in the literature since many service sourcing approaches cannot differentiate between similar offers [3, 4]. To address these problems, a tool that helps customers to freely express their requirements and to easily select the best service, has become highly recommended. The challenge of our work is to propose a fair service sourcing approach that helps clients to select the most appropriate service offer according to their intentions while giving them the ability to freely express their

M. Brambilla, T. Tokuda, and R. Tolksdorf (Eds.): ICWE 2012, LNCS 7387, pp. 307–314, 2012.

functional and QoS-based requirements using their own knowledge and language. Our approach is based on computing a score for each service offer according to its semantic closeness to the client requirements.

In this paper, we propose a new service sourcing approach which is based on semantic enabled models for client intentions and provider offers using ontologies. Based on these models, the first step of our approach consists in finding correspondences between the description of the services offered by providers and the required ones by the client. The matching step allows detecting direct and indirect correspondences between client and providers terms. The direct correspondences consist in finding semantic equivalence between these terms using a similarity measure. The indirect correspondences are automatically generated using a QoS ontology that bridges the gap between the client and the provider terms. For example, the customer can define his requirements using the expression "*download time*" while the provider defines his offer according to the "*film size*" and the "*throughput*" offered to the client. Correspondences generated in the matching phase are then stored in a matching ontology to be used in the selection process. The second step of our approach is to define a fair classification method of service offers according to their semantic closeness to the client intentions. This method computes the score of each offer based on the number of the satisfied requirements and the distance between the values proposed by the provider and those wanted by the customer for each requirement. The models used in our approach and the semantic matching process are detailed in our ontology driven approach for automatic establishment of service level agreements (ODACE SLA) [1]. In the remaining parts of this paper we detail our service sourcing process and we present how it produces more accurate results than the existing approaches.

This paper is structured as follows: section 2 provides related works about service selection. Section 3 details our semantic approach for service sourcing. Before concluding, we present a case study to illustrate that our approach produces more fair and accurate results than the existing service selection methods.

2 Related Work

There are several approaches in the literature that tackled the problem of service selection. Several criteria can be considered when selecting the best service. The selection of services can be entirely based on price [2] or on some predefined criteria for the comparison of the values of services such as security and response time [3], [4] and [5]. These criteria may not be sufficient to correctly classify the service providers and don't allow clients to freely express their own requirements based on different and high level terms. In addition, these approaches don't control the values given by providers who can give incorrect values to increase their chances to be selected. In their work [6], Comuzzi et al have defined a set of admissible values for each QoS parameter. The values given by customers and providers must belong to the admissible values. However, the authors propose to translate the values of QoS parameters into levels. This may not give a fair classification especially in the case of a large spectrum of values. In addition, this work does not allow the comparison of

linguistic terms and only the numeric values are considered. In [7], authors present a semantic web service discovery approach using the SPARQL language to evaluate preconditions and postconditions of services and check if they satisfy the required goals. Nevertheless, this approach doesn't take into account non functional properties in the selection process. In addition, authors do not provide matching capabilities especially when the client is not an IT expert.

According to this study, we conclude that the major part of the existing work doesn't provide an effective mechanism to differentiate between similar offers especially those satisfying the same number of client's constraints. In addition, these works generally omitted scoring literal terms and they are focused only on the scoring of numeric values. In our approach, we aim to use a better expression and analysis of semantics to help the clients finding the best adequate service they need. In the next section, we detail the different steps of our service sourcing approach.

3 Semantic Service Selection Approach

In this section, we detail our approach to select the most appropriate service according to the client preferences. We start by presenting our scoring method to compute the score of each provider value. Then, we detail our selection algorithm based on the defined scores. Finally, we explain the functions that compute these scores.

3.1 Service Offers Scoring Methodology

To compute the final score (FS) of each provider value, we define two types of scores:

The Satisfaction Score (SS): This score tests if the offered value of the provider satisfies the client's constraint. It allows computing the number of constraints satisfied by each provider, but it does not distinguish similar candidates and offers that satisfy the same number of the client constraints.

The Satisfaction Degree (SD): This score gives more precision to the satisfaction score according to the closeness of the values offered by the provider to the client constraints. The closer the provider value to the required one by the client is, the higher the SD will be. In our approach, we propose to grant more importance to the client constraint satisfaction than its closeness to the value of the client. For this reason, we consider that the SD shouldn't be higher than the SS. Consequently, we propose that the sum of all the SDs of each combination of provider values must not exceed 1 (which presents the maximum SS that can be assigned to a constraint value). Then, we choose a threshold that is equal to $\alpha(n_c)$ for each SD with n_c is the number of constraints required by the client. As a result, each value of SD must be between 0 and $\alpha(n_c)$ with:

$$\alpha(n_c) = \frac{1-\varepsilon}{n_c} \qquad \text{Where} \qquad 0 < \varepsilon < 1 \tag{1}$$

The final score of each provider value is the sum of these two scores. Thus, a provider value that satisfies the client constraint will have a FS between 1 and $1 + \alpha(n_c)$ and the one that doesn't satisfy the client constraint will have a FS between 0 and $\alpha(n_c)$. Consequently, if a provider doesn't satisfy any constraint, his final score will not

exceed $n_c*\alpha(n_c)$ which is equal to (1-ε). In this case, it will never have a higher score than a provider who satisfies at least one constraint and who will have a minimum score equals to 1.

3.2 Best Service Offer Selection Algorithm

The selection phase is mainly based on checking the satisfaction of the client constraints by each provider offer. It allows evaluating and selecting service providers according to their closeness to satisfy client preferences. In this section, we present our selection algorithm which allows ranking service providers on the basis of their ability to satisfy the client requirements. The first step of our algorithm is to gather all the client constraints from the intention instance. For each constraint, we consider its property, its operator and its threshold from the client ontology. Then, if the threshold value is numeric, we consider its admissible values from the QoS ontology using the correspondences generated in the matching phase. In this step, we select QoS values that have valid correspondences with the client property (we refer to $VQ_{k,min}$ and $VQ_{k,max}$ respectively, the minimum and maximum admissible values identified by the expert for the parameter associated to the k^{th} constraint of the client). A correspondence is considered valid if its certainty is above a minimum acceptance threshold. This threshold is identified using information science measures as precision, recall and F-measure [8]. After that, we retrieve all the corresponding values for each client constraint from the provider ontology.

- For the direct correspondences, we select the provider values that have valid correspondences with the client property and we try to get the value $VP_{ij,k}$ of each selected instance. $VP_{ij,k}$ denotes the value of the offer j of the provider number i which corresponds to the client constraint number k.
- For the indirect correspondence, we retrieve the QoS instances that have valid correspondences with the client property. For each selected instance found, we collect its function and its operands. The next step is to find the set of values from the provider ontology which have valid correspondences with these operands. Then, we compute the function result that presents the provider value $VP_{ij,k}$.

After retrieving all the possible values of each provider corresponding to the client constraints, we compute the final score $FS_{ij,k}$ of each value $VP_{ij,k}$ according to its closeness to the value given by the client. This final score uses the admissible values retrieved from the QoS ontology. If a constraint given by the client has no semantic (direct or indirect) correspondence with a valid term of the provider ontology, this it will be considered as an unsatisfied constraint and its final score will be equal to 0. After computing the score of each provider's value, the last step is identifying the best combination of values that gives the best score OS_{ij} of each offer O_{ij} for the provider P_i using the formula (2).

$$OS_{ij} = \frac{\sum_{k=1}^{n_c} FS_{ij,k}}{n_c * (1 + \alpha(n_c))} \quad (2)$$

These steps will be repeated for each provider to rank all the available offers according to their scores. Finally, the offer that has the best score will be selected.

3.3 Scoring Functions

In this section, we define the functions that we propose to compute the scores of provider values. We distinguish two categories: linguistic terms and numeric values. To give a fair classification method, it is necessary that both scores of linguistic terms and those of numeric values belong to the values ranges specified in section 3.1.

Computing Scores of Linguistic Terms
In our approach, we aim to identify and use the several semantic relations that can exist between terms. For this reason, we present in the following our method to compute scores of linguistic terms using similarity measures.

-To compute the SS we distinguish two possible cases:

Case 1: If the client gives a single possible term VC_k for the constraint number k (i.e. the operator is "*equal*"), we assume that the client constraint is satisfied if the degree of similarity between the client term and its corresponding provider term is greater than or equal the acceptance threshold.

Case 2: If the client gives more than one term (i.e. the operator is "*in*"), we assume that the constraint of the client is satisfied if there is at least one of the terms proposed by the client that has a similarity degree greater than or equal to the acceptance threshold with the corresponding provider term.

- The SD of linguistic terms is equal to the semantic similarity degree between the client term and the provider term multiplied by $\alpha(n_c)$ in the case where the operator is "*equal*". It will be equal to the maximum similarity degree computed between the provider term and the set of all client terms multiplied by $\alpha(n_c)$ in the case where the operator is "*in*". Note that the similarity degree must be a value between 0 and 1, so that the SD will be not above the threshold defined at section 3.1 which is equal to $\alpha(n_c)$.

Computing Scores of Numeric Values
To compute the final score of the numeric values, we suppose that:

- The SS is equal to 1 if the value of the provider satisfies the client constraint and 0 otherwise according to the required operator.

-The SD is computed according to two possible cases:

Case 1: Numeric values that have a better value direction (BVD) which can be "down" or "up". These values can be used to determine if the client prefers the minimum or the maximum value among the values offered by providers. The value "down" indicates that the SD is inversely proportional to the value of the provider. The value "up" indicates that the SD is proportional to the value of the provider. For example, "response time" has a "down" better value direction. Taking inspiration from functions defined in [4], we propose to use the formula (3).

$$SD_{ij,k} = \begin{cases} \alpha(n_c) * \dfrac{VQ_{k,\max} - VP_{ij,k}}{VQ_{k,\max} - VQ_{k,\min}} & if \quad BVD = "down" \\ \alpha(n_c) * \dfrac{VP_{ij,k} - VQ_{k,\min}}{VQ_{k,\max} - VQ_{k,\min}} & if \quad BVD = "up" \end{cases} \tag{3}$$

Case 2: Numeric values that don't have a better value direction. In this case, we adopt that the more the value of the provider is closer to the value of the client, the more its SD is bigger. We distinguish three cases according to the operator used by the client.

- If the client gives one value (i.e. the operator is "*equal*", "*greaterThan*" or "*lessThan*"). We propose to compute the SD by the formula (4):

$$SD_{ij,k} = \alpha(n_c) * (1 - \frac{|VP_{ij,k} - VC_k|}{n_1}) \quad (4)$$

With n_1=max {($VQ_{k,max}$-VC_k);(VC_k - $VQ_{k,min}$)}

- If the client gives a range of values, we propose the formula (5):

$$SD_{ij,k} = \begin{cases} \alpha(n_c) * (1 - \frac{VC_{k,min} - VP_{ij,k}}{n_2}) & if \quad VP_{ij,k} \in [VQ_{k,min}; VC_{k,min}[\\ \alpha(n_c) & if \quad VP_{ij,k} \in [VC_{k,min}; VC_{k,max}] \\ \alpha(n_c) * (1 - \frac{VP_{ij,k} - VC_{k,max}}{n_2}) & if \quad VP_{ij,k} \in]VC_{k,max}; VQ_{k,max}] \end{cases} \quad (5)$$

With n_2=max {($VQ_{k,max}$-$VC_{k,max}$);($VC_{k,min}$- $VQ_{k,min}$)}

- If the client gives two or more values (i.e. the operator is "*in*"), we propose to compute the SD using the formula (6):

$$SD_{ij,k} = \max \left\{ \alpha(n_c) * (1 - \frac{|VP_{ij,k} - VC_{k,m}|}{n_3}), 0 < m \leq NV_k \right\} \quad (6)$$

With n_3=max {($VQ_{k,max}$-$VC_{k,m}$);($VC_{k,m}$- $VQ_{k,min}$), 0<m≤NV_k, NV_k is the number of the possible client values for the k^{th} constraint}

4 Case Study

To better illustrate our approach, we consider an example in which the client specifies four constraints in his intention. He wants to download "Comedy" or "Adventure" films (C_1) from a service with a greater availability than 97% (C_2), a download time less than 10 minutes (C_3) and a price less than or equal 3 Euros per film (C_4).

Table 1. An example of provider offers scores

	O_{ij}	SS	SD	FS	OS_{ij}
Provider 1	$VP_{11,1}$=Comedy	1	0,2475	1,2475	
	$VP_{11,2}$=99%	1	0,245	1,245	0,771
	$VP_{11,3}$=9 mn	1	0,214	1,214	
	$VP_{11,4}$=3.1	0	0,143	0,143	
Provider 2	$VP_{12,1}$=Action	0	0,077	0,077	
	$VP_{12,2}$=97%	1	0,24	1,24	0,73
	$VP_{12,3}$=8.5 mn	1	0,216	1,216	
	$VP_{12,4}$=3	1	0,148	1,148	
Provider 3	$VP_{13,1}$=Documentary	0	0,037	0,037	
	$VP_{13,2}$=93%	0	0,23	0,23	0,52
	$VP_{13,3}$=8 mn	1	0,218	1,218	
	$VP_{13,4}$=2.9	1	0,153	1,153	

On the other side, we consider the offers of three providers presented in Table 1. Download time values of provider offers are computed using indirect correspondences. Table 1 shows the scores of the provider offers according to the formulas presented in section 3 of this paper. In this case study, we respectively considered these admissible values for the client constraints on "availability", "download time" and "price": $VQ_{2,min}$=0, $VQ_{2,max}$=100, $VQ_{3,min}$=1, $VQ_{3,max}$=60, $VQ_{4,min}$=1 and $VQ_{4,max}$=6 and ε is equal to 0,01. To compute the linguistic terms scores, we used WordNet::Similarity measures [9].

To better illustrate the advantages of our service sourcing approach, we have compared its results with some existing works [3] and [4]. These two works give the same results. In the existing approaches, the scores of the linguistic terms cannot be computed and indirect correspondences are not taken into account. Consequently, we considered the score of linguistic terms equal to 1 if the terms are syntactically equivalent and 0 otherwise for these approaches. Figure 1 shows that the first provider will have the highest score using our approach whereas the second provider will be selected in the other approaches. In addition, these approaches give the same score to the first and the third provider despite that the first provider gives better offers than the other providers. Consequently, we conclude that our approach gives better results than the existing service sourcing methods. In fact, it allows selecting offers according to both high level functional and non functional constraints given by the client. These constraints can be based on literal values (such as C1 in this case study) or numeric values (such as C2 and C3) using several mathematical operators. Moreover, our approach can detect indirect correspondences between intentions and offers (like C3) which are not detectable by other service sourcing works. The use of two types of scores offers a fair and accurate classification method of service providers. This classification ranks provider offers according to the number of the satisfied constraints presented by the score SS and then enhances these scores by the score SD. The final score gives better ranking precision especially when some offers satisfy the same number of constraints (the case of the providers 1 and 2 in Table 1).

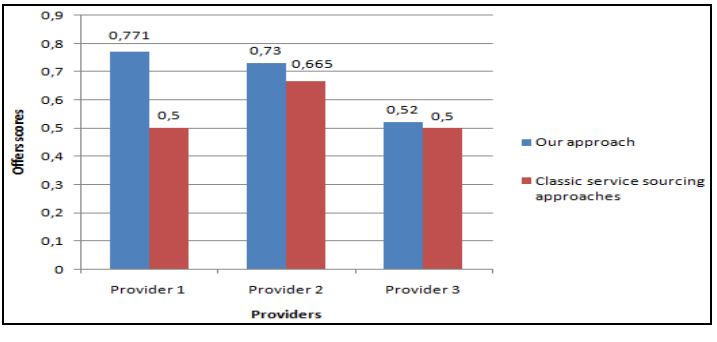

Fig. 1. Case study experimental results

5 Conclusion

Given the continuous growth of multi-service providers, the establishment of an approach that helps clients selecting the appropriate offers to their requirements is highly recommended. In fact, many offers can be distinguished even if they look

similar for non-expert clients especially when the providers use complex terms that cannot be easily understood by the clients. To achieve this goal, we defined a novel semantic approach for service sourcing. This approach helps the clients to freely express their intentions using their own language and knowledge that can be different from the provider offers. In our approach, we started by generating direct and indirect correspondences between the client and the provider terms in order to evaluate the available offers. In a second step we defined a fair and accurate scoring algorithm that can distinguish offers satisfying the same number of the client constraints and having close values. In fact, the computed scores depend on admissible ranges defined in a QoS ontology to allow a fair classification of these offers. In addition, these scores depend on the number of the satisfied functional and QoS constraints on one hand and the semantic distance between the linguistic and numeric values proposed by the provider and those required by the client on the other hand. Moreover, our approach can evaluate the provider offers using indirect QoS correspondences which are not detectable using the existing service sourcing algorithms.

As a future work, we aim to evaluate the performance of our algorithms and optimize our approach by reducing its execution time. In fact, dealing with a large number of offers can cause scalability issues. This can be avoided by deploying our algorithms on parallel environments like grids. We also intend to extend our sourcing approach to support the selection of service compositions.

References

1. Kaouthar, F., Tarak, C., Saïd, T., Mohamed, J., Khalil, D.: ODACE SLA: Ontology Driven Approach for Automatic Establishment of Service Level Agreements. IJSSOE 1(3), 1–20 (2010)
2. Lamparter, S., Ankolekar, A., Studer, R., Grimm, S.: Preference based selection of highly configurable web services. In: Proceedings of the 16th International Conference on the World Wide Web WWW 2007, pp. 1013–1022 (2007)
3. Wang, X., Vitvar, T., Kerrigan, M., Toma, I.: A QoS-Aware Selection Model for Semantic Web Services. In: Dan, A., Lamersdorf, W. (eds.) ICSOC 2006. LNCS, vol. 4294, pp. 390–401. Springer, Heidelberg (2006)
4. Zeng, L.Z., Benatallah, B., Ngu, A.H.H.: QoS-aware middleware for Web services composition. IEEE Transaction on Software Engineering 30(5), 311–327 (2004)
5. Andrikopoulos, V., Fugini, M., Papazoglou, M., Parkin, M., Pernici, B., Siadat, S.H.: QoS Contract Formation and Evolution. In: Proceedings of the 11th International Conference on Electronic Commerce and Web Technologies (EC-WEB 2010), pp. 119–130 (2010)
6. Comuzzi, M., Pernici, B.: A framework for QoS-based Web service contracting. ACM Transactions on the Web (TWEB) 3(3) (June 2009)
7. Marco, L.S., David, M., Claude, M.: Discovering Semantic Web services using SPARQL and intelligent agents. Journal of Web Semantics 8, 310–328 (2010)
8. van Rijsbergen, C.J.: Getting into Information Retrieval. In: Agosti, M., Crestani, F., Pasi, G. (eds.) ESSIR 2000. LNCS, vol. 1980, pp. 1–20. Springer, Heidelberg (2001)
9. Siddharth, P.: Incorporating dictionary and corpus information into a context vector measure of semantic relatedness. Master's thesis, University of Minnesota, Duluth (2003)