A Multi-criteria Service Ranking Approach Based on Non-Functional Properties Rules Evaluation

Ioan Toma[1], Dumitru Roman[1], Dieter Fensel[1], Brahmanada Sapkota[2], and Juan Miguel Gomez[3]

[1]DERI Innsbruck, University of Innsbruck, Austria
firstname.lastname@deri.at
[2] DERI Galway, National University of Ireland, Galway, Ireland
brahmananda.sapkota@deri.org
[3] Carlos III University, Madrid, Spain
juanmiguel.gomez@uc3m.es

Abstract. Service oriented architectures (SOAs) are quickly becoming the defacto solutions for providing end-to-end enterprise connectivity. However realizing the vision of SOA requires, among others, solutions for one fundamental challenge, namely service ranking. Once a set of services that fulfill the requested functionality is discovered, an ordered list of services needs to be created according to users preferences. These preferences are often expressed in terms of multiple non-functional properties (NFPs). This paper proposes a multi-criteria ranking approach for semantic web services. We start by briefly introducing ontological models for NFPs. These models are used to specify rules which describe NFP aspects of services and goals/requests. The ranking mechanism evaluates these NFPs rules using a reasoning engine and produces a ranked list of services according to users preferences.

1 Introduction

Service-Oriented Architectures (SOAs) are becoming a widespread solution for realizing distributed applications. Empowered by semantic technologies these solutions are evolving in what is known as Semantically Enabled Service Oriented Architectures (SESAs) [1] bringing more automatization and accuracy to various service related tasks, such as discovery, composition, ranking and selection. Among these tasks discovery, ranking and selection are core building blocks. As with most of the search products available on the market, it is not only important to determine the relevant results, but it is as well extremely important to provide the results in a relevant order. This is exactly the purpose of service ranking process, which complements the discovery process.

While problems such as discovery([6], [9], etc.) and composition([2], etc.) for Semantic Web Services have been intensively studied, the service ranking problem, has rather gathered not so much attention. However, we argue that service ranking in an important task in the overall service usage process and thus it needs to be treated accordantly. Any solution for this task is directly influence by how services are described. Three different aspects must be considered when describing a service: (1) *functional*, (2) *behavior* and (3) *non-functional*. The *functional* description contains the formal specification of

© Springer-Verlag Berlin Heidelberg 2007

what exactly the service can do. The *behavior* description contains the formal specification of how the functionality of the service can be achieved. Finally, the *non-functional descriptions* captures constraints over the previous two [3]. Among these aspects, non-functional properties need to be addressed given the high dynamism of any SOA- and SESA- based system. Furthermore, these descriptions are highly relevant for many of the service related tasks. For *ranking* especially, they are fundamental input data that need to be considered when building sorted sets of services. In this paper we present a service ranking approach which uses semantic descriptions of non-functional properties.

The paper is organized as follows: Section 2 briefly introduces our approach for modeling and attaching non-functional properties descriptions to services along with concrete examples. This solution is an integrated part of the Web Service Modeling Ontology (WSMO) [7] and its language Web Service Modeling Language (WSML) [4]. Section 3 provides a detailed description of the proposed service ranking approach. Section 4 presents initial experimental results and finally, Section 5 concludes the paper and points out perspectives for future research.

2 Non-Functional Properties

This section briefly introduce our approach on how to semantically describe NFPs of services. Furthermore concrete examples from a shipping scenario are provided. As a model and language for semantically describe services we adopt the Web Service Modeling Ontology (WSMO) [7], respectively Modeling Language (WSML) [4], due to its clean modeling solution and rule-based support.

The core of our modeling approach is a set of ontologies[1], in WSML, based on the models provided in [5]. These ontologies, provide the NFP terminology, used to specify NFPs aspects of services. Once otological models for NFPs are available, a second challenge that has to be address is how to attach NFPs descriptions to services and goals. Non-functional properties of services or goals are modelled in a way similar to which capabilities are currently modelled in WSMO/WSML [7]. Non-functional properties are defined using logical expressions same as pre/post-conditions, assumptions and effects are being defined in a capability. The terminology needed to construct the logical expressions is provided by non-functional properties ontologies (c.f. [8]).

For exemplification purposes we use services and goals from the SWS Challenge[2] Shipment Discovery scenario. We have extended the initial scenario by augmenting services description with non-functional properties aspects such as discounts and obligations[3]. The shipping services allows requestors to order a shipment by specifying, senders address, receivers address, package information and a collection interval during which the shipper will come to collect the package.

Listing 2 displays a concrete example on how to describe one non-functional property of a service (i.e Runner), namely obligations. Due to space limitations the listing contains only the specification of obligations aspects without any functional, behavioral or any other non-functional descriptions of the service. In an informal manner, the service

[1] http://www.wsmo.org/ontologies/nfp/
[2] http://sws-challenge.org/
[3] http://wiki.wsmx.org/index.php?title=Discovery:NFPUseCase

obligations can be summarized as follows: (1) in case the package is lost or damaged Runner's liability is the declared value of the package but no more than 150$ and (2) packages containing glassware, antiques or jewelry are limited to a maximum declared value of 100$.

Listing 1.1. Runner's obligations

```
namespace {_"WSRunner.wsml#",
   runner _"WSRunner.wsml#", so _"Shipment.wsml#",
   wsml _"http://www.wsmo.org/wsml/wsml−syntax/", up _"UpperOnto.wsml#"}

webService runnerService
   nonFunctionalProperty obligations
    definition
     definedBy
       //in case the package is lost or damaged Runners liability is
       //the declared value of the package but no more than 150 USD
       hasPackageLiability(?package, 150):− ?package[so\#packageStatus hasValue ?status] and
        (?status = so\#packageDamaged or ?status = so\#packageLost) and
        packageDeclaredValue(?package, ?value) and ?value>150.

       hasPackageLiability(?package, ?value):− ?package[so\#packageStatus hasValue ?status] and
         (?status = so\#packageDamaged or ?status = so\#packageLost) and
         packageDeclaredValue(?package, ?value) and ?value =< 150.

       //in case the package is not lost or damaged Runners liability is 0
       hasPackageLiability(?package, 0):− ?package[so\#packageStatus hasValue ?status] and
       ?status != so\#packageDamaged and ?status != so\#packageLost.

       //packages containing glassware, antiques or jewelry
       //are limited to a maximum declared value of 100 USD
       packageDeclaredValue(?package, 100):−
        ?package[so\#containesItemsOfType hasValue ?type, so\#declaredValue hasValue ?value] and
        (?type = so\#Antiques or ?type = so\#Glassware or ?type = so\#Jewelry) and ?value>100.

       packageDeclaredValue(?package, ?value):−
        ?package[so\#containesItemsOfType hasValue ?type, so\#declaredValue hasValue ?value] and
        ((?type != so\#Antiques and ?type != so\#Glassware and ?type != so\#Jewelry) or ?value<100).

      capability runnerOrderSystemCapability
      interface runnerOrderSystemInterface
```

Following our model for NFPs, Runner's obligations are expressed as logical rules in WSML. In a similar way other non-functional properties can be described. Further on, consider the concrete goal of shipping one package (GumblePackage) to a specified address (GumbleAddress) of a specific receiver (Gumble). A goal in WSMO is described in a similar manner to a Web service. Our concrete goal is specified in Listing 1.2.

User preferences are part of the goal. For example the user can specify which non-functional property will be used as a ordering dimension during the ranking process. In this case the ordering dimension is the obligations non-functional property (`up#nfp hasValue obl#Obligation`). Furthermore the user can specify how the results should be ordered (i.e. ascending or descending), in this case ascending (`up#order hasValue pref#ascending`), the importance of the non-functional properties e.g. for a user the price is less important than the execution time and the number of best services to be selected (`up#top hasValue "1"`). The background knowledge used during the selection and ranking process is usually extracted from the capability section of the goal.

Listing 1.2. Goal description

```
namespace {_"Goal.wsml#",
 so _"Shipment.wsml#",up _"UpperOnto.wsml#", pref _"Preferences.wsml#",
 obl _"http://www.wsmo.org/ontologies/nfp/obligationsNFPOntology.wsml}

goal Goal1
 annotations
    up#order hasValue pref#ascending
    up#nfp hasValue obl#Obligation
    up#top hasValue "1"
 endAnnotations

capability requestedCapability
 postcondition
  definedBy
   ?order[so#to hasValue Gumble,so#packages hasValue GumblePackage] memberOf so#ShipmentOrder and
   Gumble[so#firstName hasValue "Barney", so#lastName hasValue "Gumble",
   so#address hasValue GumbleAddress] memberOf so#ContactInfo and
   GumbleAddress[ so#streetAddress hasValue "320 East 79th Street",
   so#city hasValue so#NY, so#country hasValue so#US] memberOf so#Address and
   GumblePackage[so#length hasValue 10, so#width hasValue 2, so#height hasValue 3,
   so#weight hasValue 10, so#declaredValue hasValue 150] memberOf so#Package.
```

3 Ranking Services

Service Ranking is the process which generates an ordered list of services out of the candidate services set according to user's preferences. As ranking criteria, specified by the user, various non-functional properties such as Service Level Agreements (SLA), Quality of Services (QoS), etc. can be obtained from the goal description. On the service side the requested non-functional properties values are either directly specified in the service description or are provided (computed or collected) by a monitoring tool. Non-functional properties specified in goal and service descriptions are expressed in a semantic language (i.e WSML), by means of logical rules using terms from NFP ontologies.

Our solution for service ranking combines two aspects types of ranking, namely semantic ranking and multi-criteria ranking. By semantic ranking we understand any ranking mechanism which uses ontological representations of non-functional properties aspects. A multi-criteria ranking mechanism on the other hand considers multiple non-functional properties dimensions.

Non-functional properties of services and goals used in the prototype are semantically described as presented in Section 2. The logical rules used to model NFPs of services are evaluated, during the ranking process, by a reasoning engine. Additional data is required during the rules evaluation process. This data represents mainely user preferences and includes: (1) which NFPs user is interested, (2) the level of importance of each of these NFPs, (3) how the list of services should be ordered (i.e. ascending or descending) and (4) concrete instances data extracted from the goal description. The NFPs values obtained by evaluating the logical rules are sorted and the order list of services is built.

The algorithm for multi-criteria ranking based on non-functional properties is presented in listing Algorithm 1.

```
     Data: Set of services $S_{Ser}$, Goal $G$.
     Result: Order list of services $L_{Ser}$.
0.1  begin
0.2      $\Omega \longleftarrow \emptyset$, where $\Omega$ is a set of tuples $[service, score]$;
0.3      $\lambda = extractNFPs(G)$, where $\lambda$ is a set of tuples $[nfp, importance]$;
0.4      $G_{Know} = extractInstancesKnowledge(G)$;
0.5      $d = extractOrderingSense(G)$;
0.6      $\beta \longleftarrow \emptyset$, is a set of quadruples $[service, nfp, nfpvalue, importance]$;
0.7      for $s \in S_{Ser}$ do
0.8          for $nfp \in \lambda$ do
0.9              $imp = lambda.getImportance(nfp)$;
0.10             if $nfp \in s.nfps$ then
0.11                 $rule = extract(nfp, s)$;
0.12                 $nfpvalue = evaluateRule(rule, G_{Know})$;
0.13                 $\beta = \beta \cup [s, nfp, nfpvalue, imp]$;
0.14             end
0.15             else
0.16                 $\beta = \beta \cup [s, nfp, 0, 0]$;
0.17             end
0.18         end
0.19     end
0.20     for $s \in \beta$ do
0.21         $score_s = 0$;
0.22         for $nfp \in \beta$ do
0.23             $nfpvalue = \beta.getNFPValue(s, nfp)$;
0.24             $nfpvalue_{max} = max(\beta.npf)$;
0.25             $score_s = score_s + imp * \frac{nfpvalue}{nfpvalue_{max}}$;
0.26         end
0.27         $\Omega = \Omega \cup [s, score_s]$;
0.28     end
0.29     $L_{Ser} \longleftarrow sort(\Omega, d)$;
0.30 end
```

Algorithm 1. Multi-criteria ranking

First a set of tuples containing non-functional properties and their associated importance is extracted out of the goal description (line 0.3). Considering the goal example provided in Listing 2 the list contains only one non-functional property, namely *obligations*. If no importance is specified the default value is consider to be 0.5 which specify a moderate interest in the non-functional property. The importance is a numeric value ranging from 0 to 1, where 1 encodes the fact that the user is extremely interested in the non-functional property and 0 encodes the fact that the non-functional property is not of interest for the user. Further on instance data from the goal is extracted (line 0.4) and a knowledge base is created. In our example the extracted instance data containers information about the receiver, the package and the destination address. The last step in extracting relevant information for the ranking process is to identify how the results should be ordered i.e. ascending or descending (line 0.5).

Once the preprocessing steps are done, each service is checked if the requested non-functional properties specified in the goal are available in service description. In case of a positive answer the algorithm the corresponding logic rules are extracted (line 0.11) and evaluated (line 0.12) using a reasoning engine which support WSML rules (e.g. MINS[4], KAON2[5] or IRIS[6]). A quadruple structure is built (line 0.13 and 0.16) containing for each service and non-functional property the computed value and the its importance. An aggregated score is computed for each service by summing the normalized values (line 0.24) of non-functional weighted by importance values (line 0.25). The results are collected in a set of tuples, where each tuple contain the service id and the computed score(line 0.27). Finally the scores values are sorted according to the ordering sense extracted from the goal and the final list of services is returned(line 0.29).

4 Experiments

To evaluate the ranking algorithm proposed in Section 3 we have implemented it as part of the WSMX [7] execution environment. The ranking of services is performed on two NFP dimensions: *obligations* and *discounts*, but it can easily support a higher number of NFPs. The set of services used in the experiments are from SWS Challenge.

Table 1. Experimental Results

NFP/WebService	**Weasel**	**Walker**	**Muller**	**Racer**	**Runner**
Obligation	0.66	0.00	0.93	0.81	0.57
Discounts	0.0	0.23	0.85	0.47	0.64
Total Score	0.71	0.19	1.76	1.16	1.03

A set of 50 goals having the same structure with the goal presented in Section 2, but with randomly generated concrete values which influence obligations and discounts values have been used to test the algorithm. Table 1 shows the average score results obtained by running the algorithm with the given input data. An empiric comparison of sample results with ideal results shows a good behavior of our algorithm.

5 Conclusions and Future Work

In this paper a service ranking approach based on semantic descriptions of services non-functional properties was proposed. We briefly introduce our approach for modeling and attaching non-functional properties descriptions to services and goals. The proposed

[4] http://tools.deri.org/mins/
[5] http://kaon2.semanticweb.org/
[6] http://sourceforge.net/projects/iris-reasoner/
[7] http://www.wsmx.org

ranking mechanism makes use of logical rules describing non-functional properties of services and evaluates them using a reasoning engine. As a last step it builds an ordered list of services considering the values computed during the rules evaluation step.

As future work we plan to specify and implement other types of ranking approaches namely social and context-aware ranking. Further on, a set of open issues and improvements need to be addressed and integrated with the current ranking solution. These include but are not limited to: how to integrate non-functional properties values collected by monitoring tools with the service ranking, how to predict non-functional values of services, which are the best solutions to collect and incorporate user feedback and last but not least to consider trust and reputation issues.

References

1. Anicic, D., Brodie, M., de Bruijn, J., Fensel, D., Haselwanter, T., Hepp, M., Heymans, S., Hoffmann, J., Kerrigan, M., Kopecky, J., Krummenacher, R., Lausen, H., Mocan, A., Scicluna, J., Toma, I., Zaremba, M.: A semantically enabled service oriented architecture. In: WImBI 2006. WICI International Workshop on Web Intelligence (WI) meets Brain Informatics, Beijing, China (December 2006)
2. Cardoso, J., Sheth, A.P.: Introduction to semantic web services and web process composition. In: Cardoso, J., Sheth, A.P. (eds.) SWSWPC 2004. LNCS, vol. 3387, pp. 1–13. Springer, Heidelberg (2005)
3. Chung, L.: Non-Functional Requirements for Information Systems Design. In: Andersen, R., Solvberg, A., Bubenko Jr., J.A. (eds.) CAiSE 1991. LNCS, vol. 498, pp. 5–30. Springer, Heidelberg (1991)
4. de Bruijn, J., Lausen, H., Krummenacher, R., Polleres, A., Predoiu, L., Kifer, M., Fensel, D., Toma, I., Steinmetz, N., Kerrigan, M.: The Web Service Modeling Language WSML. Technical report, WSML, WSML Final Draft D16.1v0.3 (2007), `http://www.wsmo.org/TR/d16/d16.1/v0.3/`
5. O'Sullivan, J., Edmond, D., ter Hofstede, A.H.M.: Formal description of non-functional service properties. Technical report, Queensland University of Technology, Brisbane (2005), Available from `http://www.service-description.com/`
6. Paolucci, M., Kawamura, T., Payne, T., Sycara, K.: Semantic matching of web services capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 333–347. Springer, Heidelberg (2002)
7. Roman, D., Lausen, H., Keller, U. (eds.): Web service modeling ontology (WSMO). Working Draft D2v1.4, WSMO (2007), Available from `http://www.wsmo.org/TR/d2/v1.4/`
8. Toma, I., Foxvog, D.: Non-functional properties in Web services. Working draft, Digital Enterprise Research Insitute (DERI) (August 2006), Available from `http://www.wsmo.org/TR/d28/d28.4/v0.1/`
9. Verma, K., Sivashanmugam, K., Sheth, A., Patil, A.: Meteor-s wsdi: A scalable p2p infrastructure of registries for semantic publication and discovery of web services. Journal of Information Technology and Management (2004)