Recommend me a Service: Personalized Semantic Web Service Matchmaking

Anna Averbakh, Daniel Krause, Dimitrios Skoutas
L3S Research Center
Hannover, Germany
{averbakh,krause,skoutas}@l3s.de

Abstract

In the Semantic Web the discovery of appropriate Semantic Web Services for a given service request, the so-called matchmaking, is a crucial task in order to bring together Web Service provider and users in an automatic manner. While most of the current matchmaking algorithms focus on purely syntactic or semantic similarity or a combination of both (hybrid approaches), the user is not taken into account in the matchmaking process itself. Hence, specific preferences and needs of a user are not taken into account in the matchmaking process.

In this paper we show how users can be engaged in the matchmaking process by providing Web 2.0 interaction to collect user feedback. Furthermore, we present the ongoing work of the integration of collaborative filtering algorithms into the matchmaking process to generate personalized matchmaking results.

1 Introduction

In today's Web, two emerging trends can be observed: On the one hand, a growing number of Web sites adheres to Web 2.0 principles [O'Reilly, 2005] and allows users generate, share, tag, comment or rate content. Hence, applications can structure and personalize large content repositories by utilizing the power of the masses. On the other hand, more and more machine-readable data in RDF format, enriched by meta-data, is available on the Web – Sindice[1], for example, has indexed more than 50 million Web pages containing RDF data – fostering machine-machine interaction on the Web.

Both trends, as stated by Tim Berners-Lee [Berners-Lee, 2006], fit well together: Web 2.0 applications, offering simpler and interactive user interfaces, can be combined with the intelligence of the Semantic Web. A key task to bring both, Web 2.0 and Web 3.0 together is the matchmaking of Semantic Web Services: User interfaces rely on the functionality offered by Web Services and hence need to find high quality Web Services, while the matchmaking process for such Web Services can highly benefit from the interaction with the user.

The matchmaking process itself fulfills the main requirements to apply personalization: a) the available amount of Web Services is too big to be browseable and b) the requirements of users are heterogeneous. However, most of today's state-of-the-art algorithms do not take the user into account.

In this paper, we present an generic approach, that utilizes various semantic and syntactic matchmakers to apply collaborative filtering algorithms to the matchmaking process of Web Services.

2 Scenario

Alice, Paul and Bob love music and enjoy listening to music in their favorite Web 2.0 application. All of them have different favors for the kind of music: While Alice likes classics best, Paul enjoys jazz most and Bob is addicted to rock music. In order to select songs that fit best to the favors of the listeners, the Web 2.0 application invokes Web Services to receive music recommendations.

In our scenario, three different recommender services are available, that recommend music. Two of them - due to a limited datasource - can only recommend specific kinds of music, jazz music and rock music, while the third service is a general purpose music recommender service.

A non-personalized application would specify a request that matches generic music recommender services and hence would not receive the jazz and rock music recommender as first matchmaking result – even though we consider them as optimal for Paul and Bob. To receive personalized matchmaking results, an application needs to adapt the service requests according to a user's preferences. Thus, applications need to have a quite precise knowledge of the domain in order to specify appropriate requests. Furthermore, applications have to rely on properly specified and interlinked ontologies: For example, a service request for classical music recommenders can only be answered in our scenario if classical music is explicitly specified as subconcept of music – elsewise the general purpose music service could not be identified as a match.

As enabling personalized matchmaking by adapting the service request is not an optimal solution becomes more obvious if a fourth service occurs in the system that also recommends rock music but delivers much better recommendations. Both rock music recommender services have the same service description and hence cannot be distinguished by current matchmakers.

However, Alice, Paul and Bob enjoy Web 2.0 features and comment, bookmark and rate the music that they have listened to. In order to utilize these additional information about a user, a matchmaking architecture is needed that stores user feedback and utilizes it in the matchmaking process.

[1] http://www.sindice.com/

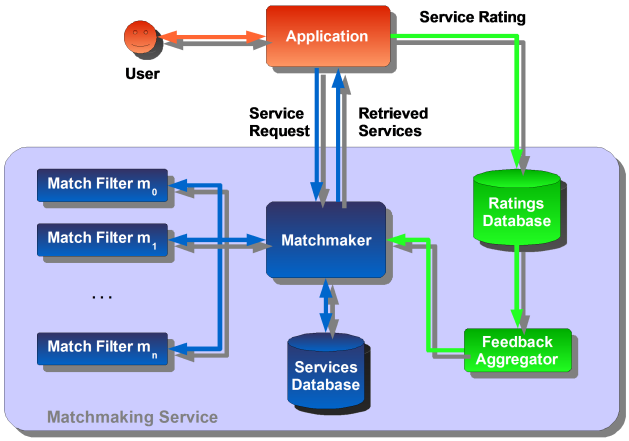

Figure 1: Personalized matchmaking service with feedback component

3 Architecture

Our system was build as a service oriented architecture with the matchmaking service (see Figure 1) as central component. This matchmaking service can by invoked by applications to receive a personalized list of matching services to a given service request. Therefore, the service invokes match filters, which implement a specific matchmaking algorithm, e.g. extended Jaccard similarity coefficient, cosine similarity etc. (see [Klusch *et al.*, 2006]) and a feedback aggregator that assigns a score to the services based on the previously given rating of the users.

After the matchmaking service returned a list of services, users can assign new rating, expressing whether the service was appropriate for the given service request or not. As the task of rating the quality of a service in respect to a given request is not trivial for a user, applications can assist the user in order to give precise feedback. For example, if an application uses services to generate music recommendations, then users can be asked whether they consider the given recommendations appropriate. Based on the assumption that services delivering high quality recommendations are better matches for this task, the application can infer the relevance of a service, and pass this information as a user rating to the matchmaking service. Such user ratings are stored in the ratings database and can be accessed by the feedback aggregator.

More detailed information about the architecture can be found at [Averbakh *et al.*, 2009].

4 Personalized Matchmaking

For personalized matchmaking, we use a domination-based matchmaking approach, as described in [Skoutas *et al.*, 2009]. This approach uses the skyline algorithm [Kossmann *et al.*, 2002] to combine multiple matchmaking metrics. Besides the existing matchmaker metrics $M_0 - M_4$ from the OWLS-MX matchmaker [Klusch *et al.*, 2006], we define an additional metric rec_x, that expresses whether a service shall be recommended to a user or not.

Assume that the collected user ratings are stored as a set $\mathcal{T} \subseteq \mathcal{U} \times \mathcal{R} \times \mathcal{S} \times F$ in the ratings database, where $\mathcal{U}$ is the set of all users that have provided a rating, $\mathcal{R}$ is the set of all previous service requests stored in the system, $\mathcal{S}$ is the set of all the available Semantic Web Service descriptions in the repository, and $F \in [0,1]$ denotes the user rating, i.e., how relevant a particular service was considered with respect to a given request (with higher values representing higher relevance). Thus, a tuple $T = (U, R, S, f) \in \mathcal{T}$ denotes that a user U considers the service $S \in \mathcal{S}$ to be relevant for the request $R \in \mathcal{R}$ with a score f.

The recommendation score rec_1 of a service s_1 and a given request r_1 for a specific user u_1 can be calculated as the average of the previous ratings from the user u_1 for service s_1 in respect to request r_1:

$$rec_1(u_1, s_1, r_1) = \frac{\sum_{(u_1,s_1,r_1,f)\in T} f}{|\{(u_1, s_1, r_1, f) \in T\}|} \quad (1)$$

However, if a user specifies a request for the first time this formula is not applicable. We can overcome this new-request problem by assuming that for similar requests a user will rate services similarly.

If $SIM_r \subseteq R$ denotes a set of services requests that are considered as similar to a given service request r and $sim(r_1, r_2) \in [0,1]$ denotes the similarity value between r_1 and r_2, rec_2 is calculated by:

$$rec_2(u_1, s_1, r_1) = \frac{\sum_{x\in X} f * sim(r_1, r_2)}{|X|} \quad (2)$$

with

$$X := \{(u_1, s_1, r_2, f) \in T : r_2 \in SIM_{r1}\} \quad (3)$$

Hence, the more similar a request r_2 is to a given request r_1, the more important is the given feedback of s_1 to r_2 for r_1.

As the amount of available Web Services grows rapidly (already today the latest OWLS test collection[2] contains more than 1000 Semantic Web Services) the user ratings - service matrix will become very sparse. Hence, the above formula will not be applicable in many cases.

To overcome the sparsity problem, we now consider also ratings from other users SIM_u, which are considered similar to the given user u_1. We consider users to be similar if they have rated services similarly. Assume that the users are represented by their rating vector, $sim(u_1, u_2)$ denotes the cosine similarity between the two rating vectors of the users u_1 and u_2. Then, the collaborative filtering approach as presented in [Shardanand and Maes, 1995] can be applied to rec_3 by:

$$rec_3(u_1, s_1, r_1) = \frac{\sum_{y\in Y} f * sim(u_1, u_2) * sim(r_1, r_2)}{|Y|} \quad (4)$$

with

$$Y := \{(u_2, s_1, r_2, f) \in T : r_2 \in SIM_{r1}, u_2 \in SIM_{u1}\} \quad (5)$$

Hence, ratings from very similar users that rated a service s_1 in the context of a given request r_2 that is very similar to the request r_1 is considered as highly relevant for the recommendation score of s_1 in respect to r_1.

5 Related Work

With the advent of Semantic Web Services like WSDL-S [Akkiraju and et. al., 2005], OWL-S [Burstein and et. al., 2004] or WSMO [H. Lausen, A. Polleres, and D. Roman (eds.), 2005], the shortcomings of keyword based search offered by seekda[3] or syntactic matchmaking performed on

[2] available at http://www.semwebcentral.org/projects/owls-tc/

[3] http://seekda.com/

WSDL files war replaced by semantic matchmaking.

Based on such Semantic Web Services, some approaches already exist about involving the user in the process of service discovery. Ontologies and user profiles are employed in [Balke and Wagner, 2003], which then uses techniques like query expansion or relaxation to better satisfy user requests. However, such an approach will not solve the task to identify which of two given services with identical service descriptions to choose. The work presented in [Xu *et al.*, 2007] focuses on QoS-based Web Service discovery, proposing a reputation-enhanced model. A reputation manager assigns reputation scores to the services based on user feedback regarding their performance. Then, a discovery agent uses the reputation scores for service matching, ranking and selection. The application of user preferences, expressed in the form of soft constraints, to Web Service selection is considered in [Kießling and Hafenrichter, 2002], focusing on the optimization of preference queries. The approach in [Lamparter *et al.*, 2007] uses utility functions to model service configurations and associated user preferences for optimal service selection. In [Dong *et al.*, 2004], different types of similarity for service parameters are combined using a linear function with manually assigned weights. Learning the weights from user feedback is proposed, but it is left as an open issue for future work. Collaborative filtering for discovering Web Service registries was presented in [Sellami *et al.*, 2009]. In [Manikrao and Prabhakar, 2005] collaborative filtering is used to re-rank the matching candidates from non-personalized matchmaking algorithms. Due to the separation of matchmaking and ranking, this approach tends to prefer weak matching popular services to well-matched services.

6 Conclusion and Future Work

In this paper we outlined by a scenario that a non-personalized one-fits-all matchmaking approach will not provide optimal service results. We introduced our architecture that enables users to influence the matchmaking process by giving feedback by well-known Web 2.0 techniques. Based on previous work, we combined different semantic and syntactic matchmaking algorithms and collaborative filtering based on user feedback by a skyline approach.

As future work, we will conduct a user study in order to evaluate the performance of our personalized matchmaking approach.

References

[Akkiraju and et. al., 2005] Rama Akkiraju and et. al. Web Service Semantics - WSDL-S. In *W3C Member Submission*, November 2005.

[Averbakh *et al.*, 2009] Anna Averbakh, Daniel Krause, and Dimitrios Skoutas. Exploiting user feedback to improve semantic web service discovery. Technical report, 2009.

[Balke and Wagner, 2003] Wolf-Tilo Balke and Matthias Wagner. Cooperative Discovery for User-Centered Web Service Provisioning. In *ICWS*, pages 191–197, 2003.

[Berners-Lee, 2006] Tim Berners-Lee. Semantic web and web 2.0. conference presentation, November 2006.

[Burstein and et. al., 2004] Mark Burstein and et. al. OWL-S: Semantic Markup for Web Services. In *W3C Member Submission*, November 2004.

[Dong *et al.*, 2004] Xin Dong, Alon Y. Halevy, Jayant Madhavan, Ema Nemes, and Jun Zhang. Similarity Search for Web Services. In *VLDB*, pages 372–383, 2004.

[H. Lausen, A. Polleres, and D. Roman (eds.), 2005] H. Lausen, A. Polleres, and D. Roman (eds.). Web Service Modeling Ontology (WSMO). In *W3C Member Submission*, June 2005.

[Kießling and Hafenrichter, 2002] Werner Kießling and Bernd Hafenrichter. Optimizing Preference Queries for Personalized Web Services. In *Communications, Internet, and Information Technology*, pages 461–466, 2002.

[Klusch *et al.*, 2006] Matthias Klusch, Benedikt Fries, and Katia P. Sycara. Automated Semantic Web service discovery with OWLS-MX. In *AAMAS*, pages 915–922, 2006.

[Kossmann *et al.*, 2002] Donald Kossmann, Frank Ramsak, and Steffen Rost. Shooting stars in the sky: an online algorithm for skyline queries. In *VLDB '02: Proceedings of the 28th international conference on Very Large Data Bases*, pages 275–286. VLDB Endowment, 2002.

[Lamparter *et al.*, 2007] Steffen Lamparter, Anupriya Ankolekar, Rudi Studer, and Stephan Grimm. Preference-based selection of highly configurable web services. In *WWW*, pages 1013–1022, 2007.

[Manikrao and Prabhakar, 2005] Umardand Shripad Manikrao and T. V. Prabhakar. Dynamic selection of web services with recommendation system. In *NWESP '05: Proceedings of the International Conference on Next Generation Web Services Practices*, page 117, Washington, DC, USA, 2005. IEEE Computer Society.

[O'Reilly, 2005] Tim O'Reilly. O'Reilly Network: What is Web 2.0, September 2005.

[Sellami *et al.*, 2009] Mohamed Sellami, Samir Tata, Zakaria Maamar, and Bruno Defude. A recommender system for web services discovery in a distributed registry environment. *Internet and Web Applications and Services, International Conference on*, 0:418–423, 2009.

[Shardanand and Maes, 1995] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating "word of mouth". In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[Skoutas *et al.*, 2009] Dimitrios Skoutas, Dimitris Sacharidis, Alkis Simitsis, Verena Kantere, and Timos Sellis. Top-k Dominant Web Services under Multi-criteria Matching. In *EDBT*, pages 898–909, 2009.

[Xu *et al.*, 2007] Ziqiang Xu, Patrick Martin, Wendy Powley, and Farhana Zulkernine. Reputation-Enhanced QoS-based Web Services Discovery. In *ICWS*, pages 249–256, 2007.