Meaningful Service Classifications for Flexible Service Descriptions

Sudhir Agarwal and Martin Junghans
Karlsruhe Institute of Technology (KIT),
Institute AIFB and KSRI,
Englerstr. 11, Karlsruhe, Germany.
Email: {sudhir.agarwal,martin.junghans}@kit.edu

***Abstract*—We present a formal underpinning for *Web service classes* by viewing them as a set of services that fulfill a logical combination of constraints on functional and non-functional properties. A hierarchy of service classes is automatically derived by their formal definition and can be exploited for an efficient service retrieval. In addition, we show in this paper how service classes can be used (i) to create service descriptions without specifying precise property values and (ii) to create service requests that can use service classes to express ranges of desired property values.**

I. Introduction

Based on the benefits of service classifications, the potential of underspecified modeling formalisms, and derived from current shortcomings we introduce our service class formalism in this work. A class formally describes a set of services that have certain service properties within a common range. The classes can be build upon functional and non-functional property constraints uniformly. Our approach provides the following benefits. (1) A formal class definition allows to automatically categorize semantically described services into given classes. The class hierarchy can be automatically derived from the definition of individual classes. Inconsistencies in existing service categorizations and contradictions between semantic service descriptions and its classification can be automatically detected and prevented. (2) Service classes can be used to express service requests. A request either formalizes requirements from scratch or alternatively reuses given class definitions. As available service descriptions can be classified automatically due to formal service class definitions, the classification of services can be pre-computed and services of a requested class can be directly retrieved, which in turn leads to more efficient service retrieval task as the search space can be reduced. (3) Service classes can be also used to create service descriptions when precise property values are not known or should not be revealed. For instance, if the precise response time of a Web service cannot be determined, a class FastService that is defined such that a service of this class feature a response time of less than a second, then the service can be classified into this class to express the responsiveness implicitly.

Our contribution can be outlined with the support of Figure 1. The bottom layer of the figure contains the interpretation of service descriptions: service instances with their executions runs (traces) and the measured values of the non-functional properties. Explicit service descriptions can be created using these models for the capability expression and the concrete value of the property availability. Service class are used to create implicit service descriptions and is exemplified by the class FastService in the middle layer of Figure 1.

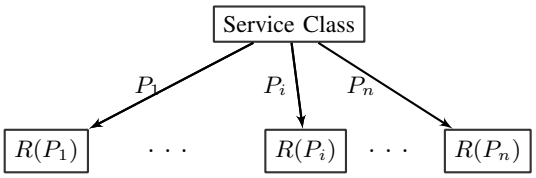

Fig. 2. Formal Property-Based Model of Web Service Classes

II. Implicit Description of Web Services Properties

The property based service model is used to formally describe Web services, including their functionality, when concrete values of the functional or non-functional properties are known and the service description modeler is willing to publish them. However, in many cases the concrete value of a service property is not known or not supposed to be mentioned explicitly. However, it may still can be possible or desired to specify a range in which the concrete value lies.

Furthermore, explicit description of property values does not support negation, which means that explicit description does now allow exclusion of properties. Therefor, since ontology languages have open world semantics, an ontology reasoner, e.g. HermiT, will not find any matching explicit service descriptions, such as OWL-S Profiles, if the request contains exclusion of some Web service property.

In this section, we introduce the notion of a *Web service class* to address above drawbacks of explicit service descriptions. We will first show formally what a Web service class means by relating it to the formal model of a Web service. Then, we will present how Web service classes can be described with OWL. Having this, we will discuss how properties of Web services can be specified implicitly with the help of classes.

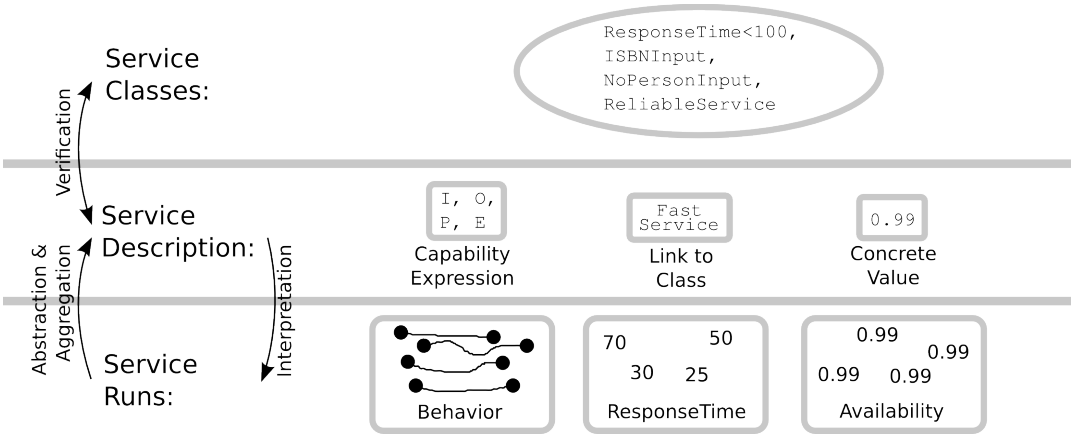

Fig. 1. Interrelation between service classes on the top layer, service description formalism using explicit and implicit values (intermediate layer), and the formal interpretation of the service model (bottom layer).

A. Web Service Classes

Definition A Web service class $\mathcal{C}$ is a finite set $\mathcal{P}$ of Web service properties, with each property $P \in \mathcal{P}$ associated with a set $R(P)$ that denotes the range for the values of the property P (refer to Figure 2).

Informally, a Web service class describes a set of Web services by specifying constraints that are satisfied by the values of functional and non-functional properties of each Web service that is a member of the Web service class. More formally, a Web service class as defined in Definition II-A describes a set of sets of (property-value) pairs (P, v) for all properties $P \in \mathcal{P}$ and for all values $v \in R(P)$. That is,

$$\mathcal{C} = \bigcup_{P \in \mathcal{P}} \{(P, v) : v \in R(P)\}.$$

In the following, we show how the range $R(\textit{functionality})$ can be defined. Recall that a service run consists of a set of inputs $\vec{i}$, a start state s^i, a set of outputs $\vec{o}$, and an output state s^o. A Web service functionality class analogously consists of a set $\mathbb{I}$ of sets of inputs, a set S^i of start states, a set S^o of output states and a set $\mathbb{O}$ of sets of outputs. A Web service functionality class $R(\textit{functionality}) = (\mathbb{I}, S^i, S^o, \mathbb{O})$ is interpreted as a set $\mathcal{L}$ of Web service executions (refer to Figure **??**) by constructing a sequence for each set of inputs $\vec{i} \in \mathbb{I}$, each start state $s^i \in S^i$, each output state $s^o \in S^o$ and each set of outputs $\vec{o} \in \mathbb{O}$. Consequently, the set of possible values of the functionality property is the set of all possible service execution sequences that respect the constraints specified in the functionality class $R(\textit{functionality})$.

The meaning of the class for a non-functional property is defined analogously according the structure of the non-functional property.

B. Description of Web Service Classes with OWL

Suppose the set of all services is denoted by $\mathcal{S}$. We begin with modeling an OWL class Service to denote $\mathcal{S}$. A subset of $\mathcal{S}$, say $\mathcal{S}_1$ with properties $\mathcal{P}$ is defined as follows. We define an OWL class $\mathsf{Service}_1$ as subclass of the class Service with $\mathsf{Service}_1 \sqsubseteq \mathsf{Service}$. For each service property $P \in \mathcal{P}$ with range $R(P)$, we define an OWL object property P with range concept $\mathsf{R(P)}$ if $R(P)$ is a set of individuals or an OWL data type property P with range $\mathsf{R(P)}$ if $R(P)$ is a data type. In both cases, the domain of the property P is set to $\mathsf{Service}_1$.

Note, that functionality is just another property in our model of Web services. That is, there is a property $\mathsf{hasFunctionality}$ with range concept $\mathsf{R(Functionality)}$ and domain concept Service. For simplicity we will use the concept $\mathsf{Functionality}$ instead of the concept $\mathsf{R(Functionality)}$. The concept $\mathsf{Functionality}$ is defined with two properties $\mathsf{hasStartKB}$ and $\mathsf{hasEndKB}$, both with range concept KB. The concept $\mathsf{Functionality}$ can be further sub-classed to specify a subset of all functionalities by sub-classing the KBs connected to it accordingly. Similarly, concepts denoting the set of values for the non-functional properties can be further sub-classed according to the need.

C. Implicit Description of Web Service Properties

As stated earlier, when a Web service description needs to exclude certain properties or include existence of properties without naming their values explicitly, then we need implicit service descriptions. Having the description formalism for Web service classes and property range classes, it is rather straight forward to achieve implicit descriptions. Since we describe Web services as well as property values as OWL instances and Web service classes as well as property ranges as OWL concepts, a Web service or a property value can be classified in a Web service class or a property range class by a OWL class member assertion axiom. Another major advantage of such a modeling technique is that it supports hybrid descriptions in the sense that some of the properties of a Web service may be described explicitly while other properties may be described implicitly. The formal semantics of OWL ensures that an OWL reasoner draws correct consequences.

Furthermore, when Web service classes are not just labels as it is the case in existing approaches, but have formal definition, it becomes possible to automatically detect inconsistencies in

service classification. For example, with existing approaches, it is possible to classify a service with an output of type *book* as a *WeatherInformationService*, even though the service does not deliver any weather information. In our approach the service class *WeatherInformationService* will have a formal description of, for example, the fact that it has only one output of type *WeatherInformation*. Therefore, when the book selling service is classified as *WeatherInformationService*, the inconsistency is automatically detected by an OWL reasoner.

D. Example

In our example from Section **??**, we have seen above how the start KB of the description of the service s explicitly names the input variables that the service has along with their relationships with each other. In order to be able to state that the service s does not have an input variable of type CreditCard, the functionality of the service s needs to be classified into a functional class. To achieve this, we first describe a subclass of the concept KB with $\mathsf{NoCreditCardInput} \sqsubseteq \mathsf{KB} \sqcap \neg\exists \mathsf{has}.\{\mathsf{CreditCard} \sqcap \mathsf{Input}\}$. Assuming ϕ denotes the start KB of the service s for which we want to describe that it does not have an input of type CreditCard, we achieve the desired description by adding the axiom $\mathsf{NoCreditCardInput}(\phi)$.

III. Conclusion and Outlook

In this paper we presented a semantic Web service modeling and matchmaking approach that exploits the benefits of formally defined service classes. We therefore based our work on available semantic Web service modeling frameworks and extended the state of the art by providing a formal model of service classes. Service classes describe services with certain functional and non-functional properties with property values in a desired range. Service class definitions provide an explicit semantics as we provided a description formalism that is based on description logics.

Based on the semantics and the common formal model of service and request descriptions, we defined a match between both and also showed how matches between descriptions and requests are computed. Finally, we completed the presented work by a prototypical implementation of our semantic discovery approach, which is a part of the larger system developed under the research project SOA4All.

Our long term goal is to establish a scalable semantic service search framework, which tightly integrates discovery and ranking of services into the search. Functional and non-functional properties are both not distinguished and likewise considered for discovery and ranking. Thus, we use preferences on both functional and non-functional properties enables service search to consider both types of properties for discovery and ranking as well. We aim to achieve efficiency and scalability by developing indexing structures. Known service matching techniques like subsume and plugin match may be exploited to develop indexing among service descriptions, since these techniques use the same formalism for service description and request. Furthermore, due to the tied coupling of discovery and ranking task, the search space that is considered for expensive semantic reasoning operations can be reduced in early stages of search. By this we aim at a very short time to retrieve first search results. Similarly to Google's Web page search, further results can be computed afterwards when the user requests them.

Acknowledgment

This work was funded by the SOA4All project (http://www.soa4all.eu) sponsored by the European Commission under EC grant number FP7-215219 and by the German Federal Ministry of Education and Research (WisNetGrid project).

References

[1] UDDI, "UDDI Executive White Paper," UDDI.org, Tech. Rep., Nov. 2001. [Online]. Available: http://uddi.org/pubs/UDDI_Executive_White_Paper.pdf

[2] R. Akkiraju, J. Farell, J. A. Miller, M. Nagarajan, A. Sheth, and K. Verma, "Web service semantics – WSDL-S," in *W3C Workshop on Frameworks for Semantics in Web Services*, 2005.

[3] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne *et al.*, "OWL-S: Semantic markup for web services," *W3C Member Submission*, vol. 22, pp. 2007–04, 2004.

[4] D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, A. Pollers, C. Feier, C. Bussler, and D. Fensel, "Web Service Modeling Ontology," *Applied Ontology*, vol. 1, pp. 77–106, 2005.

[5] W3C OWL Working Group, *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 27 October 2009, available at http://www.w3.org/TR/owl2-overview/.

[6] J. de Bruijn, D. Fensel, M. Kerrigan, U. Keller, H. Lausen, and J. Scicluna, *Modeling Semantic Web Services: The Web Service Modeling Language*. Berlin: Springer, 2008.

[7] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph, Eds., *OWL 2 Web Ontology Language: Primer*. W3C Recommendation, 27 October 2009, available at http://www.w3.org/TR/owl2-primer/.

[8] Y. Sure, S. Bloehdorn, P. Haase, J. Hartmann, and D. Oberle, "The SWRC Ontology - Semantic Web for Research Communities," in *Proc. of the 12th Portuguese Conference on Artificial Intelligence - Progress in Artificial Intelligence (EPIA 2005)*, ser. LNCS, vol. 3803. Springer, December 2005, pp. 218–231.

[9] J. O'Sullivan, D. Edmond, and A. ter Hofstede, "Formal description of non-functional service properties," *Informe Técnico FIT-TR-2005-01, Business Process Management Group, Centre for Information Technology Innovation, Queensland University of Technology*, 2005.

[10] M. Junghans, S. Agarwal, and R. Studer, "Towards Practical Semantic Web Service Discovery," in *7th Extended Semantic Web Conference*, ser. LNCS. Springer, 2010.

[11] M. Junghans and S. Agarwal, "Web Service Discovery Based on Unified View on Functional and Non-Functional Properties," in *ICSC*. IEEE Computer Society, 2010.

[12] K. Sycara, M. Paolucci, A. Ankolekar, and N. Srinivasan, "Automated Discovery, Interaction and Composition of Semantic Web Services," in *Journal of Web Semantics*, vol. 1, no. 1, Dec. 2003, pp. 27–46.

[13] K. Sycara, W. S, M. Klusch, and J. Lu, "LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace," in *in Cyberspace. Autonomous Agents and Multi-Agent Systems*, 2002, pp. 173–203.

[14] D. Martin, M. Paolucci, S. McIlraith, M. Burstein, D. McDermott, D. McGuinness, B. Parsia, T. Payne, M. Sabou, M. Solanki, N. Srinivasan, and K. Sycara, "Bringing Semantics to Web Services: The OWL-S Approach," in *SWSWPC*, ser. LNCS, J. Cardoso and A. Sheth, Eds., vol. 3387. Springer, 2004, pp. 26–42.

[15] N. Srinivasan, M. Paolucci, and K. Sycara, "Semantic Web Service Discovery in the OWL-S IDE," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences - Volume 06*. Washington, DC, USA: IEEE Computer Society, 2006.

[16] M. Á. Corella and P. Castells, "Semi-automatic semantic-based web service classification," in *Business Process Management Workshops*, ser. Lecture Notes in Computer Science, J. Eder and S. Dustdar, Eds., vol. 4103. Springer, 2006, pp. 459–470.

[17] G. Li, W. Zhang, H. Li, and J. Guo, "An efficient way to accelerate service discovery and invocation," in *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, oct. 2007, pp. 1304 – 1309.

[18] R. Lara, M. Corella, and P. Castells, "A flexible model for Web service discovery," in *1st International Workshop on Semantic Matchmaking and Resource Retrieval: Issues and Perspectives*, Seoul, Korea, September 2006.

[19] B. Benatallah, M.-S. Hacid, A. Leger, C. Rey, and F. Toumani, "On automating Web services discovery," *The VLDB Journal*, vol. 14, no. 1, pp. 84–96, 2005.

[20] J. Gonzalez-castillo, D. Trastour, and C. Bartolini, "Description Logics for Matchmaking of Services," in *KI-2001 Workshop on Applications of Description Logics*, 2001.

[21] L. Li and I. Horrocks, "A Software Framework for Matchmaking Based on Semantic Web Technology," *Int. J. Electron. Commerce*, vol. 8, no. 4, pp. 39–60, 2004.

[22] M. Stollberg, M. Hepp, and J. Hoffmann, "A Caching Mechanism for Semantic Web Service Discovery," in *The Semantic Web. 6th Int. Semantic Web Conf.*, ser. LNCS 4825, K. Aberer and et al., Eds. Busan, Korea: Springer, 2007, pp. 480–493.

[23] M. Stollberg, U. Keller, H. Lausen, and S. Heymans, "Two-phase Web Service Discovery based on Rich Functional Descriptions," in *Proc. of the 4th European Semantic Web Conf.* Springer, 6 2007.

[24] U. Keller, H. Lausen, and M. Stollberg, "On the Semantics of Functional Descriptions of Web Services," in *Proc. of the 3rd European Semantic Web Conf.*, 2006.

[25] T. Vitvar, J. Kopecký, J. Viskova, and D. Fensel, "WSMO-Lite Annotations for Web Services," in *5th European Semantic Web Conf.*, ser. LNCS 5021. Springer, 2008.

[26] J. Domingue, L. Cabral, S. Galizia, V. Tanasescu, A. Gugliotta, B. Norton, and C. Pedrinaci, "IRS-III: A broker-based approach to semantic Web services," *Web Semant.*, vol. 6, no. 2, pp. 109–132, 2008.

[27] J. O'Sullivan, "Towards a precise understanding of service properties," Ph.D. dissertation, Queensland University of Technology, 2006.

[28] K. Kritikos and D. Plexousakis, "Requirements for QoS-based Web Service Description and Discovery," *Computer Software and Applications Conference, Annual International*, vol. 2, pp. 467–472, 2007.

[29] ——, "Mixed-Integer Programming for QoS-Based Web Service Matchmaking," *IEEE Transactions on Services Computing*, vol. 2, pp. 122–139, 2009.

[30] M. Klusch, B. Fries, and K. Sycara, "OWLS-MX: A Hybrid Semantic Web Service Matchmaker for OWL-S Services," *Web Semantics*, vol. 7, no. 2, pp. 121–133, 2009.

[31] M. Klusch, P. Kapahnke, and I. Zinnikus, "Hybrid Adaptive Web Service Selection with SAWSDL-MX and WSDL-Analyzer," in *Proceedings of the 6th European Semantic Web Conference on The Semantic Web: Research and Applications*, ser. ESWC 2009 Heraklion. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 550–564.