WWW OR WHAT IS WRONG WITH WEB SERVICE DISCOVERY

Position Paper for the Workshop on Frameworks for Semantics in Web Services, Innsbruck, Austria, June 2005

Authors:

Dieter Fensel
Uwe Keller
Holger Lausen
Axel Polleres
Ioan Toma

Digital Enterprise Research Institute (DERI)
National University of Ireland, Galway, Ireland
University of Innsbruck, Austria
Email: {dieter.fensel, uwe.keller, holger.lausen, axel.polleres, ioan.toma} 'at' deri.org

ABSTRACT

Various proposals for automating the discovery of Web services are available. Most of them regard the terms *service* and *Web service* as synonymous. We believe these two terms are not equivalent and it is relevant to distinguish them and to explore their relation. We argue that this fundamental distinction is necessary in order to achieve scalable and realistic Semantic Web service Discovery. Thus, we advocate a proper conceptual model following the distinction of these notions. Our model takes into account Goals, Abstract Goals, Web services and Services, in order to enable efficient and scalable discovery which matches the expectations of service requesters.

INTRODUCTION

The Internet and the World Wide Web are tremendous success stories and have changed the way we publish and communicate information in our modern society. Web services now start to add a new level of functionality on top of the existing Web and transform them from a place where we share and find data to a place where we find and share dedicated services and functionalities too. Finding suitable Web services that help to achieve a certain goal is widely considered as a key task in a such shared global marketplace of services. Semantic description of what is provided by a Web service allows the automation of the process (or parts of it) and therefore dynamically adapting software systems using a service-oriented architecture. An efficient solution of the discovery problem allows for a cost-effective construction of software systems from pre-existing components whereby single elements in the system's architecture can be dynamically exchanged. Various proposals for automating the discovery of Web services are available (e.g. [Paolucci et al., 2002], [Li and Horrocks, 2003]). These often regard the terms *service* and *Web service* as synonymous. We believe these two terms are not equivalent and, furthermore, it is relevant to distinguish them and to explore their relation, as done in [Preist, 2004] and [Baida et al., 2004]. We argue that this fundamental distinction is necessary in order to achieve scalable and realistic Semantic Web service Discovery.

The need for such distinction becomes more evident if we think about a prototypical e-business scenario: a user wants to get a specific service that provides some real-value for him. Web services are technological means for accessing or specifying services offered by some specific provider. In a sense a Web service is an access point to services of a particular provider. Users are not specifically interested in Web services but rather in the services that can be delivered by a specific provider. Clients think in terms of services whereas providers advertise sets of services they are able to offer to their clients (using a technical entity Web service). Hence, the description of a Web service is the smallest unit of advertisement for providers.

Consequently we advocate a proper conceptual model for a discovery process based on the distinction between the two notions and the distinction between goals, abstract goals, Web services and services, in order to enable service discovery. We make a clear-cut distinction between Web service discovery and service discovery, where we focus on the former in this position statement: This paper presents our understanding of the terms *service* and *Web service* by means of a simple example and their distinction and interrelation. Given this distinction, we propose a conceptual framework for discovery and briefly discuss the advantages of the proposed model.

SERVICE AND WEB SERVICE

The notion of *service* is used in different communities and even within the same communities in various ways as it has been pointed out in [Preist, 2004]. Many people have a different understanding of the term *service* which causes a lot of confusion and makes it hard to compare different work directly. For example, in the business community a *service* is seen as *a business activity that often results in intangible outcomes or benefits* [Baida et al., 2004], while in computer science the terms *service* and *Web service* are often regarded as interchangeable to name *a software entity accessible over the Internet*.

In this Section we want to clarify what precisely we understand by the two notions *service* and *Web service* and thus implicitly what kind of entities we aim to semantically describe and locate.

Service

Following [Preist, 2004] we consider a service as *a provision of value to a client in some domain*. A service represents the kind of entity, a user is interested in and that he wants to have discovered. As an example, let us consider a user who wants to book a train ticket from Innsbruck to Munich on a given date. The service he is looking for, is the provision of a train ticket with the specified constraints. Such provision is independent on how the supplier and the provider interact, i.e., it does not matter whether the requester goes to a train tickets office or uses the train Web site to book his trip.

Web service

A Web service is defined in [Preist, 2004] as *a computational entity accessible over the Internet (using Web service standards and protocols)*. Coming back to our previous example, a railway company might provide a software component accessible via Web service standards, i.e., a Web service to request the booking of a trip. Thus, the Web service is an electronic means by which a client is able to request a specific service from a provider, but not the service itself. Therefore, we understand the term *Web service* as a means to request a service over the Internet, described using agreed standards.

Web service descriptions

As introduced before, a Web service is used by clients to request the actual provision of a service fulfilling some requester need. In order to identify an appropriate Web service for a specified requester need, the Web service must describe what services it is actually able to provide. However, while the requester needs will be often concrete e.g. travelling by train from Innsbruck to Munich on a concrete date, a Web service will often be by providers used to offer a (potentially large) set of related services e.g. booking of trips from a given train company. Even though we would expect a provider to adequately describe the set of services that can be requested (using a specific Web Service), two observations make this expectation rather unrealistic:

- **Information volume**: describing all the services that can be requested using a given Web service will, in many cases, imply the consideration of the whole provider database during discovery e.g. a Web service to book trips from a given train company will have to describe all the train connections, from all origins to all destinations, and on all dates that the train company can provide. The information volumes involved in such a scenario are rather unmanageable for reasoning systems today. A scalable discovery process thus can not exclusively be based on very precise descriptions and logical reasoning at the same time. Some sort of abstraction in descriptions is needed to simplify both, reasoning as well as the description of all services to be advertised by a provider.
- **Dynamics**: the set of services accessible via a Web service in general is dynamic over time. For example, the set of train trips that can be booked using an train company Web service will depend on seat availability, which will change over time. Maintaining an accurate list of the services provided by the Web service would imply updating the Web service description every time some dynamic condition changes. Even if the description of a Web service functionality does not have to be literally changed , at least some interaction with the providers then is needed during the discovery process to achieve accurate discovery results. Such a communication can be expected to be rather expensive (in contrast to standard computation steps during discovery) and hence should be kept at a minimum in a scalable process.

Given the observations above, we do not expect Web services to describe the complete set of services they can provide. Instead, we expect Web services to describe an abstraction of the set of services that can be requested i.e. a simpler, static characterization of the kind of services that can be accessed via the Web service. For example, the train company Web service from our example will declare that it can be used to request the booking of train trips from any destination in Austria to any other destination in Austria or Germany, not detailing all the possible concrete itineraries and dates, and independently of dynamic conditions such as seat availability or weather conditions. We call such static characterization of the services provided by a Web service, an *Web service description*. The concrete service the requester is looking for will be given by a *service description*.

Further we see semantic annotations of Web services as important descriptions that make automation of Web service discovery possible. Having semantically meaningful and machine processable descriptions allow computers to support discovery and turn it into a powerful and cost-effective process. These annotations naturally lead to the notion of Semantic Web services, which are semantically self-described Web services. In order to find concrete services provided by a particular Web service, we still have directly interact with the Web service. Semantic Web services do not replace Web services but they only help to find the latter ones. They do not provide semantic descriptions for each individual concrete service. If such an assumption holds, Web services are not longer necessary, at least for the discovery process, which in this case requires a huge reasoning effort.

EXAMPLE FOR SEMANTIC WEB SERVICE DISCOVERY

To illustrate our position on discovery we use an example within the travelling domain. Assuming someone wants to buy a ticket for a trip from Innsbruck to Munich on the 30th of July 2005 at noon, with current Web service technology he would know about and consult the Web service of the Austrian railway company and query them if they have the particular transportation service available. The company might in turn offer a slightly different service, i.e. a train departing 12:35 and the customer agrees on that and the purchase is performed.

In our model of discovery a Web service provider does not duplicate the functionality of the Web service and describe all possible concrete services (i.e. all train schedules and prices), but only an abstraction of it. In the example chosen the provider would advertise to offer all passenger railway transportation services within Austria and some services between Austria and Germany. In order for the requester to find the Web service of the Austrian railway his concrete desire needs to be abstracted to a goal. The locations will be abstracted to countries and the timing information be dropped, resulting in an abstraction of the form "transportation service from Austria to Germany". With this goal the requestor is able to automatically find suitable Web service provider that may solve his desire. This process is called Web service discovery. The following process of interaction of a discovery engine with the single Web service providers to determine the subset of the relevant providers whose Web services actually are able to deliver the requested service is called service discovery and not in the scope of this paper.

CONCEPTUAL MODEL FOR THE DISCOVERY PROCESS

Research in expert systems revealed an interesting pattern on how to relate explanations to given observations in a diagnosis task. In [Clancey, 1985] a pattern called *Heuristic Classification* (see Fig.1) has been extracted from an analysis of various rule-based expert systems for a diagnosis task. The pattern contains three subsequent inference steps that together form a complex diagnosis process:

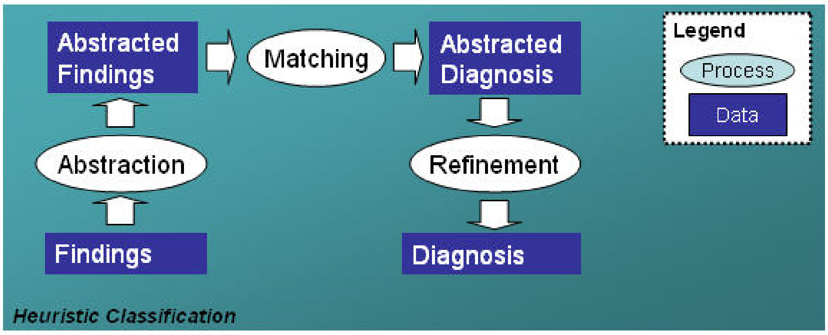

Figure 1. Problem-solving pattern: Heuristic Classification

Abstraction.
Abstraction is the process of translating concrete description of a case into features that can be used for classifying the case. For example, the name of a patient can be ignored when making a diagnosis, his precise age may be translated into an age class, and his precise body temperature may be translated into the finding "low fever". The process is about extracting classification relevant features from a concrete case description.

Matching.
Matching is the process of inferring potential explanation, diagnoses, or classifications from the extracted features. It matches the abstracted case description with abstract categories describing potential solutions.

Refinement.
Refinement is the process of inferring a final diagnosis explaining the given findings. This process may include the acquisition of new features describing the given case. However, it is now the potential solution that guides the acquisition process of these features.

As the latest step indicates, the entire process can be executed in an iterative fashion. Instead of acquiring all potential findings an initial set can be used to derive intermediate potential explanation that can be further used to guide the next iteration of the process. We propose to use the problem-solving pattern *Heuristic Classification* as the underlying conceptual model for a clean and adequate definition of the process of discovery in the Semantic Web service domain. We believe that scalable and workable approach to discovery of services requested by client has to the same pattern.

The pattern *Heuristic Classification* can be applied for the definition of a discovery process in a natural way. The relation is illustrated in Fig. 2.

Abstracting goals from user desires: Goal Discovery

Users may describe their desires in a very individual and specific way that makes immediate mapping with service descriptions very complicated. Therefore, each service discovery attempt requires a process where user expectations are mapped on more generic goal descriptions. Notice that this can be hidden by the fact that a discovery engine allows the user only to select from predefined goals. However, then it is simply the user who has to provide this mapping i.e. who has to translate his specific requirements and expectations into more generic goal descriptions. This step can be called *goal discovery* , i.e., the user or the discovery engine has to find a goal that describes (with different levels of accuracy) his requirements and desires. In the current literature on service and web service discovery this step is mostly neglected.An example of such a user desire would be to buy a train ticket from Innsbruck to Munich, on the 30th of July 2005 at leaving Innsbruck at noon.. It can be seen that this is a very concrete and detailed desire, while goals are intended to be generic and reusable, e.g. buy train tickets or buy train tickets in Europe. Therefore, a mapping from the user desire to generic goals becomes necessary.

Matching Goals and Web services: Web service discovery.

Web service discovery is based on matching abstracted goal descriptions with semantic annotations of web services. This discovery process can only happen on an ontological level, i.e., it can only rely on conceptual and reusable descriptions. For this, two processes are required: a) the concrete user input has to be generalized to more abstract goal descriptions, and b) concrete services and their descriptions have to be abstracted to the classes of services a web service can provide. We believe that this twofold abstraction is essential for lifting web service discovery on an ontological level that is the prerequisite for a scalable and workable solution for it.

Refining the result of Web service Discovery to Services: Service Discovery

Service discovery is based on the usage of web services for discovering actual services. Web service technology provides automated interfaces to the information provided by software artifacts that is needed to find, select, and eventually buy a real-world service or simply find the piece of information somebody is looking for. Service discovery requires strong mediation and wrapping, since the specific needs of a choreography of a web service have to be met in order to interoperate with it. Notice that automatization of service discovery defines significant higher requirements on mediation than web service discovery, as it also requires protocol and process mediation. In a sense, the role of web service discovery can be compared with the role of an internet search engine like Google, and service discovery with the process of extracting the actual information from the retrieved web sites.

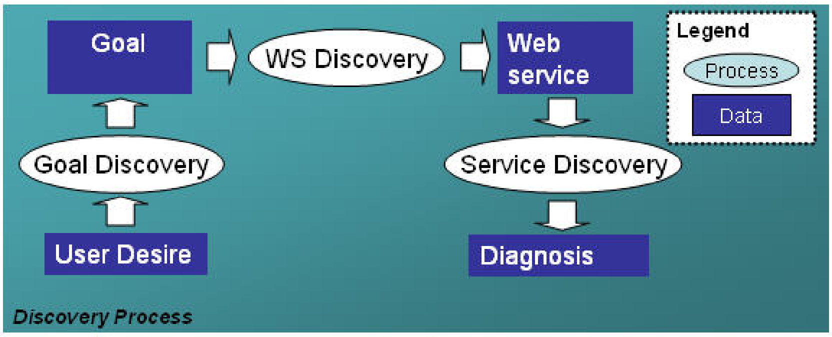

Figure 2. A conceptual model for the discovery process

RELATED WORK AND SUMMARY OF POSITION

The automatic discovery of services currently is a very popular research topic. Many approaches that address this topic in multi-agent systems or description logic based reasoning settings were proposed (cf.[Li and Horrocks, 2003], [Paolucci et al, 2002], [Benatallah et al, 2003], [Verma et al, 2004], [Sycara et al, 2002]). However they all failed to provide a viable solution to the discovery problem due to the lack of a clear conceptual model and an in our opinion incorrect interchangeable usage of service and Web service concepts. Service discovery and Web service discovery are intermixed, and complete and correct descriptions of the services are wrongly assumed. Furthermore, most of these approaches envision a logical reasoning mechanism over the huge set of service descriptions which we doubt to be efficient and scalable.

Our approach, on the other hand clearly distinguished between a service and a Web service. Services are concrete provision of value in a specific domain. Web services on the other hand are simply service discovery engines, providing the means to find services. Further we see semantic annotations of Web services, also known as Semantic Web services, as important descriptions that make automation of Web service discovery possible. They are not meant replace Web services, instead they can be seen as means to find the latter ones. In consequence, Semantic Web services help to find the right Web services, that are further used to find concrete services.

Based on all previous distinctions we have proposed a complete conceptual model for the automatic discovery of services inspired from the problem-solving method: *heuristic classification*. As part of this model, three distinct steps were identified: *goal discovery*, which is about abstracting user needs into abstract and reusable goals, *Web service discovery*, which is about matching an abstract goal against Web service descriptions, i.e. abstracted description of functionality and finally *service discovery*, which is about using Web service technology to agree with a provider on a concrete service that is to be delivered.

REFERENCES

[Baader et al., 2003] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (eds.): The Description Logic Handbook, Cambridge University Press, 2003.

[Baida et al., 2004] Z. Baida, J. Gordijn, B. Omelayenko, H. Akkermans: A Shared Service Terminology for Online Service Provisioning, Proceedings of the Sixth International Conference on Electronic Commerce (ICEC04), Delft, The Netherlands, 2004.

[Benatallah et al, 2003] B. Benatallah, M-S. Hacid, C. Rey, and F. Toumani. Request rewriting-based web service discovery. In The Semantic Web - ISWC 2003, pages 242-257, October 2003.

[Clancey, 1985] W. J. Clancey: Heuristic Classification, Artificial Intelligence 27(3), pp. 289-350, 1985

[Li and Horrocks, 2003] L. Li and I. Horrocks. A Software Framework for Matchmaking Based on Semantic Web Technology. In WWW'03, Budapest, Hungary, May 2003.

[Paolucci et al, 2002] M. Paolucci, T. Kawamura, T. Payne, and K. Sycara. Semantic Matching of Web Service Capabilities. In ISWC, pages 333–347. Springer Verlag, 2002.

[Preist, 2004] C. Preist. A Conceptual Architecture for Semantic Web Services. In Proceedings of the International Semantic Web Conference 2004 (ISWC 2004), November 2004.

[Verma et al, 2004] K. Verma, K. Sivashanmugam, A. Sheth, and A. Patil. Meteor-s wsdi: A scalable p2p infrastructure of registries for semantic publication and discovery of web services. Journal of Information Technology and Management, 2004.

[Sycara et al, 2002] K. Sycara, S. Wido®, M. Klusch, and J. Lu. Larks: Dynamic matchmaking among heterogeneous software agents in cyberspace. Autonomous Agents and Multi-Agent Systems, pages 173-203, 2002.

ACKNOWLEDGMENTS

The work is funded by the European Commission under the projects DIP, Knowledge Web, InfraWebs, SEKT and ASG; by Science Foundation Ireland under the DERI-Lion project; by the FIT-IT (Forschung, Innovation, Technologie - Informationstechnologie) under the projects RW2 and TSC. The authors would like to thank to all the members of the WSMO, WSML and WSMX working groups for their advice and input to this document

$Date: 2005/04/29 17:52:24 $