ConQo – A Context- And QoS-Aware Service Discovery

Iris Braun, Anja Strunk, Gergana Stoyanova, Bastian Buder
TU Dresden, Department of Computer Science, Chair of Computer Networks
01062 Dresden, Germany
{iris.braun, anja.strunk, gergana.stoyanova, bastian.buder}@inf.tu-dresden.de

ABSTRACT
Due to the increasing number of Web Services, which provide similar functionality, the non-functional properties are becoming more important during the selection of the best available service. Non-functional properties describe Quality of Service (QoS) as well as context of service execution. Although there are many approaches considering only QoS or context during service discovery and selection, there is a lack of systems taking both non-functional categories into account. In this paper we introduce ConQo - our approach for a context- and QoS-aware service discovery and selection.

KEYWORDS
Semantic Web Services, Service Discovery, Quality of Services (QoS), Context

1. INTRODUCTION
In order to find services fulfilling the client’s needs, a discovery mechanism is needed, that looks for appropriate services by comparing the client’s requirements with the descriptions of available services. Functional properties describe the service’s behavior as well as all information necessary to use the service. Such information are the technical interface and the address of the service endpoint. Based on the functional description the discovery component performs a matching algorithm that filters those services providing the functionality the client has requested for. The non-functional properties provide further details of the quality (QoS) and context of service execution. QoS refers to how well a service performs its behavior and is encompassing performance and network-related characteristics as well as price models and costs. Context information is based on the results of the service’s behavior. Such criteria are for example the price of the hotel booked by a hotel reservation service or the location of a printer managed by a printer service. In contrast to the QoS attributes, context information highly depend on the service domain. Based on the non-functional properties the discovery service will perform a ranking that sorts the services according to the degree of their compliance with the client’s non-functional requirements.

This paper introduces ConQo, a discovery framework considering QoS as well as context information. The remainder of the paper is organized as follows: First we analyze some related work in the field of semantic service description and discovery focused on QoS and context support. Section 3 contains the architecture of our framework and describes the service description and matching mechanisms in detail. The article concludes with final remarks and an outlook at future work.

2. RELATED WORK
In order to discover services the ability to describe the service’s functional and non-functional properties is necessary. Due to the lack of semantics in WSDL new ontology based service description languages were designed. The most important semantic description languages are OWL (W3C, 2004) and WSML (W3C, 2005). Both approaches provide ontologies (OWL-S, WSMO) to describe the functional aspects of services.

Most semantic matchmakers only compare functional properties and requirements. One approach for QoS-based discovery is WSMO-QoS (Vu et al., 2006). For optimizing the matchmaking process they perform a semantic categorization of service descriptions and user queries into different classes based on the similarity
among them. Given this categorization information, the semantic matchmaking is only applied to the user query and the set of service descriptions belonging to the same class. For each service description and user request, they generate a corresponding characteristics vector using the available information in its inputs, outputs, precondition, postcondition (IOPP) and QoS specification. They utilize Bloom filters to efficiently determine the membership of a certain concept group to a service class. The ranking is based on a reputation-based trust management mechanism to evaluate the actual QoS of the services. Approaches like OWL-Q (Kritikos, Plexousakis, 2007) und OWL-QoS (Zhou et al., 2005) define description language and reasoning algorithm, but do not propose a complete environment for semantic Web Service discovery.

Unfortunately, context descriptions are not included in OWL-S or WSMO so far, but stand-alone approaches like Context Ontology Language (CoOL) (Krummenacher et al., 2007) were developed. CoOL is a generic description language for any kind of context information and uses the Aspect Scale Context Model (ASC), but Web Service descriptions and matchmakers based on CoOL are not provided so far. Another related approach is the Context Based Service Composition Framework (CB-SeC) (Mostefaoui et al., 2003), where services are incrementally filtered and ranked according to the evaluation of the contextual information, in order to optimize the composition of Web Services. For the representation of context information Comprehensive Structured Context Profiles (CSCP) are used (Held et al., 2002), that expresses information by means of session profiles. Within the DIANE project (Klan, 2006) another approach for context-aware service discovery, selection and usage was developed. The DIANE middleware is based on the semantic service description language DSD (DIANE Service Description) (Klein et al., 2005) and builds up mechanisms for completely automatic semantic service matching, selection and binding. In summary, we can investigate that there are many specialized approaches for semantic service discovery based on functional or non-functional properties but there is no solution that bundles context- and QoS-based discovery together.

3. THE CONQO APPROACH

Our architecture is based on the WSMO execution environment WSMX and the DIP project (see Fig.1).

Fig. 1 Architecture of ConQo based on DIP QoS-Discovery Component of WSMX

The service descriptions based on WSML are managed in the Web Service Repository. Providers can use the Provider Interface to publish and update the descriptions of their services. These descriptions contain
both, functional as well as non-functional (QoS, Context) properties. As we mentioned in the Section 2, there are two prevailed approaches to describe the service’s properties. We decided to use WSMO because it is the only one which provides the necessary environment to publish, find and search Web Services. WSMO already provides the concepts to define the functional aspects of services and requirements. To describe the QoS properties we reuse the QoS upper ontology of the DIP project (Vu et al., 2006). In order to make this description more explicit to the reader we will use an example scenario. Assume we want to print a document, search for a print service. Such a service has QoS properties like price or availability. Furthermore some printer specific context information can be modeled, which refer to the result of the service, the printed document, e.g. color depth and resolution.

Due to the lack of describing context in WSMO, we defined our own upper ontology for context information called WSMO-Context. The challenge was to create generic concepts which can be reused in different use cases. We identified two types of context information: Measurable data with a certain range and a certain unit. The resolution in dpi and the queue length of the printer are e.g. measurable context information of our print service. Non-measurable data are constants like the quality (laser or inkjet) and the color depth of the printed documents. Another orthogonal classification of context information is the differentiation between static and dynamic context. The last-mentioned context values can change during run-time, e.g. the length of the queue of the printer.

Based on these generic concepts, context domains (domain specific ontologies like „PrinterContextOntologie.wsml”) can be defined and used in service descriptions as well as in goal descriptions to formulate offers and needs in form of instances of these concepts. Table 1 shows the classification of context information used in this ontology.

<table>
<thead>
<tr>
<th>context</th>
<th>static</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>non-measurable</td>
<td>color (bw/color)</td>
<td>resolution</td>
</tr>
<tr>
<td>measurable</td>
<td>quality (laser/inkjet)</td>
<td>queue</td>
</tr>
</tbody>
</table>

Table 1 Classification of context information

In addition to the requirements of a user, providers can define environmental requirements in their service descriptions. So it is possible to demand from the service user a minimal color depth of his display or a minimal bandwidth for using a service. These requirements can also be of both categories - context and QoS.

Fig. 2 shows the description of the print service with the concepts of WSMO-QoS and WSMO-Context. For registration and management of additional ontologies a Management Interface is provided.

Web Service requesters are interacting with the Web Service Requester Interface to send a goal description to the Discovery Component of the DIP project for searching in the Web Service Repository. The required non-functional properties are described in the same manner as in the service description shown in Fig. 2. For matching the axiom GoalSatisfied of the DIP project is used. The matching operations defined within the axiom are not only equal or not equal but also less than and greater than.

The result of the matching process is a list of all services which satisfy the functional and non-functional requirements described in the goal. Due to our extended domain ontologies, the discovery component supports QoS- and context aware matching now.

After this matching the ranking of the remaining set of services is performed by our modified Ranking component. At first, for getting a better performance, this component verifies all mandatory parameters. The services which do not fulfill these essential requirements get a negative ranking and will be removed from the set of applicable services. Subsequently follows the real ranking algorithm: It compares each service with all other services in the list and evaluates every parameter. For each parameter the service has a better value as the other service, it gets 1 or 2 points, depending on the value is significantly better. “Better” in this meaning could stand for “less than” as well as “greater than”. To define the adequate relational operator for each parameter the concepts describing the operator are linked to the concept of the parameter. Should the service completely fulfill or exceed the user requirements and the compared service does not fulfill them an additional point is given. The reached points for one context or QoS parameter will be multiplied with the given weight described in the goal. Than the ranking component accumulates the points for all parameters.
and all comparisons with the other services in the list and creates a final score for each service. In case of a non-measurable value (e.g. color) solely one point is given, if the service fulfills the requirement.

At the end, the response of the user request is a ranked list of services which fulfill the users QoS and context requirements. Now, the user can invoke the best available service. Unless an appropriate service was found, a detailed list of all services which fulfill the functional requirements will be presented to the requester, supplemented with the information which non-functional parameters do not fulfill the requirements.

The current prototype is entirely written in Java. In order to create an interoperable system, we use SOAP/HTTP as protocol for all communication between the components. For visualization we implemented JSP-Clients for both interfaces. The Requester Client supports formulating user queries and goals with the assistance of a form-based GUI. Providers can also use a web-based Provider Interface to publish and update their WSML-based service descriptions in the Web Service Repository.

4. CONCLUSION

In this paper we have proposed ConQo as a new approach for context- and QoS-aware discovery. Our contribution is the enhancement of semantic service discovery and selection by taking into account Quality of Service as well as contextual information. The discovery consumes a WSMO-Goal and matches this with the registered Web Services described with WSMO, WSMO-QoS und WSMO-Context. ConQo integrates the DIP QoS component in a practical environment and extends the ranking component with new ranking strategies and algorithms. Contrary to the original ranking algorithm of the DIP project not the reputation data are used but the actual values from the service description. This is very important for dynamic, measurable values which can change during run-time. These values may be periodically actualized by a monitoring component we have developed in another project context and which will be integrated in the near future.

In conclusion, it should be mentioned that this was a preliminary work so there are some aspects that should be the focus of further investigations. Our next steps will be the evaluation of the matchmaking as well as the ranking algorithms. Furthermore, the architecture has to be extended with an invocation handler such that the best service can be invoked immediately. Thus, the user sends a request for a given functionality, the discovery service searches, selects and invokes an appropriate service and sends the results back to the user.
5. REFERENCES