Service retrieval based on behavioral specification

Daniela Grigori, Mokrane Bouzeghoub
Prism, Universite de Versailles Saint-Quentin en Yvelines
45 avenue des Etats-Unis, 78035 Versailles Cedex, FRANCE
{Daniela.Grigori, Mokrane.Bouzeghoub}@prism.uvsq.fr

Abstract

The capability to easily find useful services (software applications, software components, scientific computations) becomes increasingly critical in several fields. Current approaches for components retrieval are mostly limited to the matching of their inputs/outputs. To go beyond the limits of these approaches, a substantial effort has been done by different works on semantic web and ontologies by exploiting more knowledge on the semantics of the components. However, this effort still remains insufficient and does not fulfill user needs as many functional or quality aspects are hidden within the specification of components behavior. In this paper we argue that, in many situations, the component discovery process should be based on the specification of this behavior, that is the process model which describes each component. The idea behind is to develop matching techniques that operate on process models and allow delivery of partial matches and evaluation of semantic distance between these matches and the user requirements. Consequently, even if a service satisfying exactly the user requirements does not exist, the most similar ones will be retrieved and proposed for reuse by extension or modification. To do so, we reduce the problem of service behavioral matching to a graph matching problem and we adapt existing algorithms for this purpose.

Keywords: web services, services retrieval, behavioral matching

1. Introduction

The capability to easily find useful services (software applications, software components, scientific computations) becomes increasingly critical in several fields. Examples of such services are numerous:

- Software applications as web services which can be invoked remotely by users or programs. One of the problems arising from the model of web services is the need to put in correspondence service requesters with service suppliers, especially for services which are not yet discovered or which are new, taking into account the dynamic nature of the Web where services are frequently published and removed.
- Programs and scientific computations which are important resources in the context of the Grid, sometimes even more important than data [6]. In such a system, data and procedures are first rank classes which can be published, searched and handled. Thus, the scientists need to retrieve procedures with desired characteristics, to determine if a required calculation was already carried out and whether it is more advantageous to carry out it again or to retrieve data generated previously.
- Software components which can be downloaded to create a new application. To reduce the development, test and maintenance costs, a fast solution is to re-use existing components.

In all these cases, users are interested in finding suitable components in a library or collection of models. User formulates a requirement as a process model; his goal is to use this model as a query to retrieve all components whose process models match with a whole or part of this query. If models that match exactly do not exist, those which are most similar must be retrieved. For a given task, the models that require minimal modifications are the most suitable ones. Even if the retrieved models have to be tailored to the specific needs of the task, the effort for the tailoring will be minimal.

The next section presents existing approaches for service retrieval and shows their drawbacks. In section 3 we show how the behavioral matching is reduced to a graph matching problem and presents the algorithm which constitute our background. Section 4 shows how this algorithm can be used for composite service discovery. Finally section 5 presents ongoing work and conclusions.

2. Related work

Currently, the algorithms for Web services discovery in registers like UDDI or ebXML are based on a search by key words or tables of correspondence of couples (key-value). Within the framework of the semantic Web, description logics were proposed for a richer and precise formal description of services. These languages allow the definition of ontologies, such as for example DAML-S, which are used as a basis for semantic matching between a declarative description of the required service and descriptions of the services offered ([10, 2, 1]). In [10, 1], a published service is matched with a required service when the inputs and outputs of the required service match (are equal or a generalization of the type) the inputs and outputs of the published service. In [2] a query language for services is porposed which allows to find services by specifying conditions on the activities which compose them, the exceptions treated, the flow of the data between the activities. In [7] independent filters are defined for service retrieval: the name space, textual description, the domain of ontology that is used, types of inputs/outputs and constraints. The approach presented in [4] takes into account the operational properties like execution time, cost and reliability.

In the context of the Grid [6], the research of procedures is based on a high-level language which expresses the relations between procedures and data used in input and produced as output.

Service retrieval based of key words or some semantic attributes is not satisfactory for a great number of applications. The tendency of recent work is to exploit more and more knowledge on service components and behavior. The need to take into account the behavior of the service described by a process model was underlined by several researchers [16, 13, 2, 18, 12].

In [2], in order to improve precision of web service discovery, the process model is used to capture the salient behavior of a service. The greater expressiveness of process models, as compared with keywords, offers the potential to increase substantively retrieval precision. The authors of [13] argues that the matchmaking based on service input and output is not sufficient as some output data may be produced only under certain internal conditions. Thus, they propose an algorithm that matches output data taking into account the process structure, for instance conditional branching. Authors of [16] underlines the importance of including behavior aspects in matchmaking process in the B2B environment and mention it as a future work. In [12] a model for dynamic service aggregation is presented; the authors stress also the capability to automatically verify the behavioral compatibility of various processes as a requirement in electronic marketplaces. The work presented in [18] deals with the equivalence of two processes modeled using Petri nets. It is supposed that partners discover each other by searching in business registry, and then they agree on a protocol. The paper poposes a method to verify the compatibility between the agreed protocol and the process existing in the enterprise. We take a different approach, by allowing to find a partner that is compatible (fully or partially) to an existing enterprise process.

Our objective is to propose an approach for service retrieval based on behavioral specification. To the best of our knowledge there is not another approach allowing to retrieve services having similar behavior.

3. Background

Web services interface definition is complemented by conversation model and composition model. Conversation protocol describes the observable behavior of a web service by specifying constraints on exchanged messages order for correct interaction with the service. The composition model allows to integrate web services according to the specification of an explicit process. For both models (composition and conversation), most of existing proposals (standard and research models) are graph based. For this reason, we choose to base our service retrieval approach on process graphs. A process is represented as a directed graph, whose nodes are activities. Edges have associated transition conditions expressing the control flow dependencies between activities.

An activity having more than one incoming control edge is a join activity; it has an associated join condition (that is a boolean exprssion on the transition conditions of the incoming edges). Activities have a type and can be atomic or composed. Activities associated to web services consume input data elements and produce output data elements. They specify also the name of the operation.

In summary, a process model is represented by a directed attributed and labeled graph whose nodes are activities and whose edges represent control flow constraints.

Using graphs as representation formalism for both user requirements and service models, the service matching problem turns into a graph matching problem. We want to compare the process graph representing user requirements with the model graphs in library. The matching process can be formulated as a search for graph or subgraph isomorphism. However, it is possible that it does not exist a process model such that an exact graph or subgraph isomorphism can be defined. Thus, we are interested in finding process models that have similar structure, if models that have identical structure do not exist. The error-correcting graph matching integrates the concept of error correction (or inexact matching) into the matching process [14, 3].

In order to compare the graphs in the library (that will be called model graphs in the following) to the graph expressing user requirements (called input graph) and decide which model is more similar to the input, it is necessary to define a distance measure for graphs. Similar to the string matching problem where edit operations are used to define the string edit distance, the subraph edit distance is based on the idea of edit operations that are applied to the model graph. The graph edit operations are used to alter the model graphs until there exist subgraph isomorphism to the input graph. A certain cost is assigned to each graph edit operation. The subgraph edit distance from a model to an input graph is then defined to be the minimum cost taken over all sequences of edit operations that are necessary to obtain a subgraph isomorphism. It can be concluded that the smaller the subgraph distance between a model and an input graph, the more similar they are.

The subgraph isomorphism detection is based on a state-space search by means of an algorithm similar to A* [14]. Different algorithms have been proposed forerror-correcting graph matching in order to reduce the computation complexity (see for example [8, 9]).

4. Semantic composite web service discovery

In this section we illustrate the use of the graph matching algorithm presented above for semantic composite web service discovery. Suppose that user needs to develop a new composite web service. He specifies the composition model and then he will try to find in the library similar web services or fragments that could be composed to develop the new web service.

In order to allow a semantic match, we suppose that the following semantic information is attached to each service (the composite web service and each web service component) similar to [15]:

- Name of operation. In the process definition, an activity attribute, called operation-concept is added which allows users to search for operations based on ontological concepts. For instance, operations buyTicket and cancelTicket are mapped to ontological concepts TicketBooking and TicketCancellation in the TravelService Ontology.

- Input and output parameters of activities. Using ontologies to annotate input and output elements brings user requirements and service advertisements to common conceptual space and helps to apply reasoning mechanism to find a better match. The semantics implied by these parameters, which are known only to provider of the service, can be made explicit. For example, the input Travel Details and output Confirmation are mapped to ontological concepts TicketInformation and ConfirmationMessage, respectively.

  The mapping of parameters to ontological concepts could be done manually or semi-automatically. In [5] an algorithm is presented that clusters operation parameters names from WSDL descriptions into meaningful concepts. In [11] a framework for semi-automatically marking up web service descriptions with ontologies is proposed.

- preconditions and effects. Each activity may have a number of preconditions and effects. The preconditions are some logical conditions, which must be true for executing the activity. Effects are changes in the world after the execution of the operation. These preconditions and effects are mapped to ontological concepts. Similarly, transition conditions associated to edges are mapped to ontological concepts.

User enters web service requirements as a template constructed using ontological concepts. The template and existing process models are transformed in graphs. In this way, the problem of service matchmaking is transformed into a problem of graph matching. If user defines input, output parameters and operation name for the new composite web service, then a first filter could be applied for components retrieval based on these properties. The behavioral matching will be applied either to the set of services retrieved in the first step or independently.

We use the algorithm forerror-correcting subgraph isomorphism to retrieve the most similar models. For each attribute of a service, the cost function of substituting an attribute value has to be defined. These attributes being concepts in the ontology, the cost function is the distance between these concepts in the ontology. We use Wu and Palmer similarity measure [17] to calculate this distance. Given a tree, and two nodes a,b of this tree, we first of all find their deepest (in terms of depth in the tree) common ancestor c. The similarity measure is computed as follows:

$S_{WP}(a,b) = 2 \times Depth(c)/(Depth(a) + Depth(b) + 2 \times Depth(c))$

The distance (cost) will be:

$d(a,b) = 1 - S_{WP}(a,b)$

Each service is characterized by following attributes : operation name, input, output, pre-conditions and effects. For the error-correcting algorithm the cost function for each graph operation has to be defined. The cost for deletion/insertion of an edge and an vertex can be set to constants. The cost for substituting a label and its attributes is defined as follows. Given a label l_1 and attributes $a_1, a_2, \cdots a_n$ and another label l_2 and attributes $b_1, b_2, \cdots b_m$, the cost for substituting l_1 with l_2 is computes as follows. If l_1 and l_2 are not identical then the substitution cost is set to the corresponding constant. If, on the other hand l_1 and l_2 are identical and n=m, the substitution cost C_S is defined to be the weighted mean of distances between $a_1, a_2, \cdots a_n$ and $b_1, b_2, \cdots b_n$, i.e.,

$C_S = \sum d(a_i, b_i) \times w_i / \sum w_i$

Applying the error-correcting subgraph isomorphism algorithm to compare user requirements with the model graphs allows to find the most similar models in the library. The algorithm outputs also the transformations needed, for exemple adding an edge, or changing the order of two nodes. In a second stage, if some models cover disjoint subgraphs of user model, a composition of these models can be proposed.

5. Conclusion and Future Work

In this paper we proposed a solution for process retrieval based on behavioral specification. By using a graph representation formalism for services, we proposed to use a graph error correcting matching algorithm in order to allow an inexact matching. We are developing a prototype that implements the behavioral matchmaking as a web service. This web service takes as input the graph representations of two processes and calculates the degree of similarity between them and outputs also the transformations needed to transform one process into the other. It invokes the service that matches services based on functional requirements (services considered as black boxes) for which methods were proposed in the literature. We are working also on defining and implementing a set of process management operators useful for process retrieval.

We plan to extend the algorithm to tackle the situation when one node in the first graph corresponds to a subgraph of the second graph. This situation appears when the first service has a single operation (activity) to achieve certain functionality, while in the second service the same behavior is achieved by receiving several messages. For instance, a service requires to receive the purchase order as well as shipping preferences in one message, while the second one needs two separate messages for this purpose (sendShippingpreferences and submitOrder).

While in this paper we dealt with the semantics aspects of behavioral matchmaking, we did not address the operational aspects. The graph isomorphism computation being a NP-complete problem, in our future work we will try to apply constraints and heuristics to cut down the computational effort to a manageable size.

References

[1] B. Benatallah, M. Hacid, C. Rey, and F. Toumani. Semantic reasoning for web services discovery. In *WWW2003 Workshop on E-Services and the Semantic Web*, 2003.

[2] A. Bernstein and M. Klein. Towards high-precision service retrieval. In *Int. Semantic Web Conference (ISWC)*, 2002.

[3] H. Bunke. Recent developments in graph matching. In *15th Int. Conf. on Pattern Recognition*, pages 117 – 124, 2000.

[4] J. Cardoso and A. Sheth. Semantic e-workflow composition. *Journal of Intelligent Information Systems*, 21:191–225, 2003.

[5] L. Dong, A. Halevy, J. Madhavan, E. Nemes, , and J. Zhang. Similarity search for web services. In *VLDB*, 2004.

[6] I. Foster, J. Voeckler, M. Wilde, and Y. Zhao. Chimera: A virtual data system for representing, querying and automating data derivation. In *14th Conf. on Scientific and Statistical Database Management*, 2002.

[7] T. Kawamura, J. De Blasio, T. Hasegawa, M. Paolucci, and K. Sycara. A preliminary report of a public experiment of a semantic service matchmaker combined with a uddi business registry. In *1st International Conference on Service Oriented Computing (ICSOC)*, 2003.

[8] B. Messmer. *Graph Matching Algorithms and Applications*. PhD thesis, University of Bern, 1995.

[9] B. Messmer and H. H. Bunke. A new algorithm for error-tolerant subgraph isomorphism detection. *IEEE Trans. PAMI*, 1998.

[10] M. Paolucci, T. Kawamura, T. R. Payne, and K. Sycara. Semantic matching of web services capabilities. In *First International Semantic Web Conference (ISWC)*, 2002.

[11] A. Patil, S. Oundhakar, A. Sheth, and K. Verna. Meteor-s web service annotation framework. In *WWW Conference*, 2004.

[12] G. Piccinelli, G. Di Vitantonio, and L. Mokrushin. Dynamic service aggregation in electronic marketplaces. *Computer Networks*, 2(37), 2001.

[13] S. S. Bansal and J. M. Vidal. Matchmaking of web services based on the DAML-S service model. In *Int. Joint Conference on Autonomous Agents and Multiagent Systems*, pages 926–927, 2003.

[14] L. G. Shapiro and R. M. Haralick. Structural descriptions and inexact matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 3, 1981.

[15] Sivashanmugam, K. Verma, A. Sheth, and J. Miller. Adding semantics to web services standards. In *International Conference on Web Services (ICWS)*, 2003.

[16] D. Trastour, C. Bartolini, and J. Gonzalez-Castillo. A semantic web approach to service description for matchmaking of services. In *Int. Semantic Web Working Symposium (SWWS)*, 2001.

[17] Z. Wu and M. M. Palmer. Verb semantics and lexical selection. In *32nd Annual Meetings of the Associations for Computational Linguistics*, 1994.

[18] J. Zdravkovic and P. P. Johanesson. Cooperation of processes through message level agreement. In *Int. Conf. On Advanced Information Systems Engineering (CAISE)*, 2004.