

Service Retrieval Based on Behavioral Specifications and Quality Requirements

Daniela Grigori, Verónica Peralta, Mokrane Bouzeghoub

Laboratoire PRiSM, Université de Versailles
45, avenue des Etats-Unis, 78035 Versailles Cedex, France
{Daniela.Grigori, Veronika.Peralta, Mokrane.Bouzeghoub}@prism.uvsq.fr

Abstract: The capability to easily find useful services becomes increasingly critical in several fields. In this paper we argue that, in many situations, the service discovery process should be based on both behavior specification (that is the process model which describes each composite service) and quality features of services. 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. To do so, we reduce the problem of service behavioral matching to a graph matching problem and we adapt existing algorithms for this purpose. The matching algorithm is extended by a flexible quality evaluation procedure which checks whether a given service is worth to be delivered or not.

1 Introduction

The capability to easily find useful services becomes increasingly critical in several fields. Examples of such services are numerous: (i) software applications as web services which can be invoked remotely by users or programs; (ii) programs and scientific computations which are important resources in the context of the Grid; (iii) software components which can be downloaded to create a new application.

In all these cases, users are interested in finding suitable services in a library or in a catalog of services. Service retrieval may be based on their inputs/outputs and their constraints (pre and post conditions) or on some high level attributes which characterize, at some extent, their functional semantics. Recent publications have demonstrated that this approach is not sufficient to discover relevant services [2]. Many services with different semantics may have the same inputs/outputs or the same constraints, therefore leading to a lack of relevance of the retrieval process. To go beyond the limits of this approach, a substantial effort has been done by different works on semantic web and ontologies by exploiting more knowledge on the semantics of the services [1][7]. High level conceptual graphs and assertions intend to capture more meaning of supplied services. However, this effort still remains insufficient and does not fulfill user needs as many functional or quality aspects are hidden within the specification of services behavior.

In this paper we argue that, in many situations, the process of service discovering should be based on both behavior specification and quality features of services.

Behavior specification abstracts the functional semantics of the components while quality features describes non-functional aspects, i.e. their added-value, constraints in terms of performance, reliability, data consistency, data freshness, etc. The idea behind is to develop matching techniques that operate on process models as well as the associated quality features 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.

In the approach presented in this paper, we reduce the problem of service behavioral matching to an adorned graph matching problem, where the graph represents the functional process of a service and the adornment represents quality constraints. Our approach is based on existing algorithms on graph matching [6] which are adapted to workflow diagrams. The matching algorithm is extended by a flexible quality evaluation procedure which checks whether a given service is worth to be delivered or not. A set of similarity metrics based on functional and non-functional requirements are defined.

The rest of the paper is organized as follows: Section 2 presents a graph representation of the process models, shows how the behavioral matching problem can be reduced to a graph matching problem and defines a structural similarity measure. Section 3 presents the data quality evaluation principle and defines a qualitative similarity measure that is added to the matching approach. Finally, section 4 presents ongoing work and conclusions.

2 Behavioral Matching

In this section, we present our approach to service retrieval based on their behavioral models. Behavioral models are process models that describe user requirements as well as provided services. We assume that the formal models are workflow models although the approach can easily be adapted to other formal models such as Petri nets or state chart diagrams, provided that it is a uniform model. After giving the preliminary definitions, we describe the principles of service matching algorithm.

2.1 Background

Most of existing proposals (standard and research models) for process specification 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 vertices are activities or data repositories. Activities associated to web services consume input data elements and produce output data elements which may persist in repositories. There are two types of edges: (i) *control edges* that have associated transition conditions expressing the control flow dependencies between activities, and (ii) *data edges* representing data flow between activities.

Using graphs as representation formalism for both user requirements and service models, the service retrieval problem turns into a graph matching problem. We want to compare the process graph representing user requirements with the process graphs

in the 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 [6].

In order to compare the graphs in the library (that will be called *catalog graphs* in the following) to the graph expressing user requirements (called *query graph*) and decide which model is more similar to the latter, 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 subgraph edit distance is based on the idea of edit operations that are applied to the catalog graph altering it until there exists a subgraph isomorphism to the query 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 edit 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*. Different algorithms have been proposed for error correcting graph matching in order to reduce the computation complexity (see for example [6]).

2.2 Service Retrieval Based on Behavioral Matching

In this section we begin by showing how the error-correcting subgraph isomorphism algorithm can be used for behavioral matchmaking. Then we define a similarity measure based on the subgraph edit distance allowing ranking models in the library.

Suppose that a user needs to develop a new composite web service. He specifies his composition model as a query graph that he submits to the service retrieval system to find in the library similar web services or fragments that could be composed to develop his application or a new web service. If we assume that existing services are represented as graphs (e.g. workflows), 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 for error correcting subgraph isomorphism to retrieve the most similar models. For the error correcting algorithm, the cost function for each graph operation has to be defined. The costs are application dependent and reflect the likelihood of graph distortions. The more likely a certain distortion is to occur the smaller is its cost.

The cost for deletion/insertion of an edge and a vertex can be set to constants. The cost for substituting a label and its attributes is defined as follows. If the labels are not identical then the substitution cost is set to the corresponding constant. If they are identical and they have the same number of attributes, the substitution cost is defined to be the weighted mean of distances between the corresponding attributes. For each attribute of a service, the cost function of substituting an attribute value has to be

defined. In [4], we showed how this cost can be defined if service attributes are annotated with ontological concepts. Attributes being associated with concepts in the ontology, the cost function is the distance between these concepts in the ontology.

In order to rank the catalog graphs, a similarity measure has to be defined. The total distance between the two graphs can be defined as the sum of the subgraph edit distance and the cost of inserting the vertices of the query graph not covered by the error correcting subgraph. The similarity measure is the inverse of this distance. For more details on behavioral matching see [4].

3 Data Quality Evaluation

The relevance of the service retrieval process can be enhanced by adorning service behavior with quality features. Section 2 has shown how the retrieval process is reformulated as a graph matching problem. However, if the matching algorithm is only based on functional requirements, the resulting services may not necessarily fulfill user expectations in terms of performance, reliability and consistency. Additionally, if some of these services are data providers, freshness and accuracy of this data could be of high importance to their users. This section goes further to improve such matching by introducing quality measures which help to discriminate between several matches.

The idea behind this approach is to adorn catalog graphs with quality features (e.g. costs, delays, data freshness, data accuracy) which allow estimating the quality of the data that can be produced by the graphs. The calculated quality values are compared to the quality constraints of the query graph (also used as adornments).

In this section we present an example that illustrates how we utilize these adornments for evaluating a concrete quality factor: *data freshness* and then we describe the matching approach using a similarity metric based on data quality.

3.1 Evaluating Freshness Within the Adorned Graph

Data freshness is a quality factor which is very important in many data centric applications. Decision making may not be relevant if it is based on stale data. Data freshness can be measured as the time passed since the creation of data or as the time elapsed since the last delivery of data; an extensive study of this has been done in [3].

The freshness of result data depends on the freshness of input data but also on the amount of time the service needs for executing its activities. The latter depends on the processing cost of activities and on the synchronization delays that can exist between their executions due to control flow constraints. Then, the quality features used as graph adornments mainly are:

- *Processing cost*: It is the amount of time, in the worst case, that an activity needs for reading input data, executing and building output data.
- *Synchronization delay*: It is the amount of time passed between the executions of two consecutive activities.
- *Input actual freshness*: It is a measure of the actual freshness of source data.

We propose an evaluation algorithm that estimates the freshness of result data based on the graph adornments. The algorithm traverses the graph following the sense of the edges, calculating the freshness of the data resulting from each activity. The principle is the following (algorithm pseudocode can be read in [5]):

- If an activity A reads external input data, result data freshness is calculated adding the actual freshness of input data and the processing cost of the activity.
- If an activity A has one predecessor B, result data freshness is calculated adding the freshness of the data produced by B, the synchronization delay between B and A and the processing cost of A.
- In the general case, if activity A has several predecessors, the freshness of data coming from each predecessor (plus the corresponding synchronization delay) should be combined and added to the processing cost of the activity A. If activity A also reads external data, actual freshness of input data should also be considered and combined. Typical combination functions are maximum, average or weighted average, but other user-specific functions can be considered, for example, for ignoring some predecessor because its data is stable (ex. country names).

3.2 Using Quality Requirements in Behavioral Service Retrieval

In this section we extend the behavioral matching procedure in order to take into account quality requirements. We distinguish two scenarios: (i) the query requirements express constraints, i.e. the catalog graphs that do not fulfill quality constraints should not be returned, and (ii) the quality requirements express expectations, i.e. a catalog graph that does not fulfill quality expectations but offers the most similar values can be returned. In the former, the user prefers retrieving a graph that is structurally less similar (and then having more development cost) but satisfying his quality requirements. In the latter, quality expectations are more flexible, and can be balanced with structural similarity. In the following we discuss both approaches.

Matching under Quality Requirements: In this scenario, quality requirements are expressed as quality thresholds that catalog graphs must verify to be retrieved. The retrieval steps are: (i) match the catalog graphs in order to obtain the isomorphic subgraphs and calculate the structural similarity measure; (ii) evaluate data freshness and eliminate the candidates that do not achieve freshness requirements; and (iii) rank the candidate graphs according to their structural similarity and retrieve the best one.

Matching with Quality Expectations: In this scenario, the model graphs can be ranked according to a similarity measure that takes into account both structural similarity and freshness expectations. A qualitative similarity measure can be defined in order to express the degree in which freshness expectations are achieved. Depending on the application, the similarity measure can be defined in different ways.

An example of such a measure is given in the following formula:

$$S_Q = (\text{ExpectedFreshness} - \text{ActualFreshness}) / \text{ExpectedFreshness} \quad (1)$$

The formula calculates the difference between expected and actual freshness values and normalizes it dividing by the expected values. When freshness expectations are achieved, the similarity is a positive value that will act positively in the global

similarity measure. When freshness expectations are not achieved the similarity is a negative value having the opposite effect.

Having defined a qualitative similarity measure, we can build a global similarity measure that combines structural and qualitative ones in a weighted sum. The weights should indicate user preference for structural over quality criteria. The retrieval steps are: (i) match the catalog graphs in order to obtain the isomorphic subgraphs and calculate the structural similarity measure; (ii) evaluate data freshness and calculate the qualitative similarity measure; and (iii) rank the candidate graphs according to their global similarity and retrieve the best one.

4 Conclusion

In this paper we proposed a solution for service retrieval based on behavioral specification and quality requirements. First, we proposed to use a graph error correcting matching algorithm in order to allow an inexact matching. Then, we showed how the quality factors can be used in the matchmaking process.

We implemented the behavioral matchmaking as a web service. The prototype takes as input the graph representations of two services and calculates the degree of similarity between them, also returning the sequence of transformations needed to transform one service into the other. We also implemented a prototype of a quality evaluation tool, which takes as input the graph representation of a service and returns a measure of its data quality. Current work addresses the integration of the two functionalities in the perspective of the ideas presented in this paper.

While in this paper we dealt with the semantics aspects of behavioral matchmaking, we did not address the operational aspects. The graph matching computation is 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 Benatallah, B., Hacid, M.S., Rey, C., Toumani, F.: Semantic Reasoning for Web Services Discovery. In Proc. of the Workshop on E-Services and the Semantic Web (ESSW), Hungary (2003)
- 2 Berstein, A., Klein, M.: Towards High-Precision Service Retrieval. In Proc. of the 1st Int. Semantic Web Conference (ISWC), Italy (2002)
- 3 Bouzeghoub, M., Peralta, V.: A Framework for Analysis of Data Freshness. In Proc. of the 1st Int. Workshop on Information Quality in Information Systems (IQIS), France (2004)
- 4 Grigori, D., Bouzeghoub, M.: Service Retrieval Based on Behavioral Specifications. In proc. of Int. Conf. Of Service Computing, USA (2005)
- 5 Grigori, D., Peralta, V., Bouzeghoub, M.: Service Retrieval Based on Behavioral Specifications and Quality Constraints. Technical report, University of Versailles (2005)
- 6 Messmer, B.: Graph Matching Algorithms and Applications. PhD Thesis, University of Bern (1995)
- 7 Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic Matching of Web Services Capabilities. In Proc. of the 1st Int. Semantic Web Conference (ISWC), Italy (2002)