IJCSMS International Journal of Computer Science & Management Studies, Vol. 12, Issue 02, April 2012
ISSN (Online): 2231 –5268
www.ijcsms.com

COMPATIBLE SERVICE RETRIEVAL USING IMPROVED SIMILARITY MEASURE

N. Arunachalam [1], N. Balasubramanian[2], S. Gokulakrishnan[3], V. Sathish[4]

[1]Assistant Professor, Information Technology
Sri Manakula Vinayagar Engineering College Puducherry, India

[2, 3, 4] Student, Information Technology,
Sri Manakula Vinayagar Engineering College Puducherry, India
bala91.kpn@gmail.com

Abstract

Now-a-days retrieving suitable services become a prominent need for the user. However available service retrieving mechanism uses the compatible similarity between the services so that user can get likely homogeneous services. This paper work proposes a document based search which uses cosine measure for comparing WSDL files for retrieving similar services. The development of Web Services and Web Based Application are made into clusters by their characteristics, and these clusters are used for the document based search and it has the advantage of reducing the complexity by suppressing the number of services during search. When the number of services increases the complexity of the retrieval of the services also increased.

Keywords: *Cosine measure, QoS, URBE, WSDL, and Document based Search.*

INTRODUCTION

During the recent years, the number of publicly available Web services has been increasing steadily. An important step in enabling the Service-Oriented Architecture (SOA) paradigm is the ability of service and SBA developers (simply referred to as developers from now on) to be able to retrieve potentially relevant services. In particular, in this work we focus on the task assigned to developers to identify at design-time which activity is to be performed and to discover and select the Web services closest to their requirements. Furthermore, it is also necessary to be able to identify and replace services participating in a service composition at run-time. In this sense, service retrieval, frequently referred to also as service discovery, is a critical step for reusing existing services while developing other services and SBAs. Several approaches have been proposed in the literature for Web services retrieval.

We can distinguish between two categories of solutions: registry-based and ontology-based ones. Ontology-based approaches do not usually consider the structure of the Web service interfaces and they require additional effort in the form of annotations to produce a service description. For these reasons, and despite the effectiveness of ontology-based approaches, we focus on a registry-based solution. In particular, we use as the starting point for our approach the URBE matchmaker . URBE is an approach for service retrieval based on the evaluation of similarity between Web service interfaces. In URBE, each Web service interface is defined in Web Services Description Language (WSDL); a matchmaking algorithm combines the analysis of their structures with the similarity of the used terms in order to retrieve relevant services for purposes of replacing a service.

IJCSMS International Journal of Computer Science & Management Studies, Vol. 12, Issue 02, April 2012
ISSN (Online): 2231 –5268
www.ijcsms.com

URBE performs on average better when compared to approaches like and for this reason it was selected as the baseline for our work. Both URBE and similar solutions consider not only the structure but also the semantics of candidate services. They do not however take into account the purpose of each element in the service description with respect to service compatibility. Service compatibility refers to the property of preservation of interoperability for internalized changes to one or both interacting parties (service provider or consumer), or equivalently, of the capacity for replacing one service with another (also referred to as substitutability and replaceability).

Different elements in the service description have different effects on interoperability: adding for example an operation to a WSDL document does not have any effect on existing clients of the service; removing an operation however may affect them dramatically. For this purpose, the compatibility is preserved as long as the properties of covariance of output and contra-variance of input are preserved. This is a property that is not considered in the matchmaking algorithms discussed in URBE and similar approaches.

To this effect, in this work we aim to combine service retrieval with service compatibility with the goal of improving the matchmaking of URBE. The new matchmaking algorithm takes into account not only the interface structure and term similarity, but also the importance of each element for service compatibility.

As a result of this synergy, retrieved services are not only similar to the required service, but additionally, a minimum effort is demanded from the developers in order to be able to use this service on the composite service or SBA side. Furthermore, the updated URBE implementation is shown to perform better both in terms of precision and average response time with respect to the older version.

RELATED WORKS

Research on Web services covers a multitude of issues that are involved throughout the life-cycle of a Web service or a Service-Based Application (SBA). Among them, service description and service composition have attracted a great deal of attention from researchers. By conducting an extensive literature review, we identified two major interrelated problems: the lack of formal specifications for service compositions (or even for atomic Web services) and the inability of automated Web service composition approaches (especially the ones that employ AI planning techniques) to simultaneously satisfy requirements such as QoS awareness, dynamicity and scalability in an effective way.

Due to the web services' heterogeneous nature, which stems from the definition of several XML based standards to overcome platform and language dependence, web services have become an emerging and promising technology to design and build complex inter-enterprise business applications out of single web-based software components. To establish the existence of a global component market, in order to enforce extensive software reuse, service composition experienced increasing interest in doing a lot of research effort. This paper discusses the urgent need for service composition, the required technologies to perform service composition. It also presents several different composition strategies, based on some currently existing composition platforms and frameworks, re-presenting first

IJCSMS International Journal of Computer Science & Management Studies, Vol. 12, Issue 02, April 2012
ISSN (Online): 2231 –5268
www.ijcsms.com

implementations of state-of the-art technologies, and gives an outlook to essential future research work.

Service compatibility can be distinguished in two dimensions: Horizontal compatibility (or service interoperability) and vertical compatibility (also known as substitutability or replaceability). Horizontal compatibility or interoperability of two services expresses the fact that the services can participate successfully in an interaction as service provider and service consumer. The underlying assumption is that there is at least one context (configuration of the environment, resource status and message exchange history) under which the two services can fulfill their roles.

On the other hand, vertical compatibility or substitutability (from the provider's perspective) or replaceability (from the consumer's perspective) of service versions expresses the requirements that allow the replacement of one version by another in a given context. Compatibility is traditionally further decomposed into backward and forward. A definition of forward and backward compatibility with respect to languages in general, and message exchanges between producers and consumers in particular. Forward compatibility means that a new version of a message producer can be deployed without the need for updating the message consumer(s). Backward compatibility means that a new version of a message consumer can be deployed without the need for updating the message producer. Full compatibility is the combination of both forward and backward compatibility.

The usual approach for defining what constitutes a compatible change to a service is to enumerate all possible compatibility preserving changes to a service description, usually a WSDL document. The allowed changes essentially define the type of delta between two service versions for which the versions are compatible and they are usually expressed in a guideline style.

1) Add (optional) message data types.
2) Add (new) operation.
3) Add (new) port type.

Any other modification like the removal or any kind of modification to an operation element is strictly prohibited, as is the modification of the message data types (with the exception of addition of optional data types). This guideline based approach is easily applicable and requires minimum support infrastructure and for this reason is widely accepted.

On the other hand, it is also very restrictive and, even more importantly, it depends on service developers for deciding what is compatible and what is not and acting accordingly. Even if these rules are codified and embedded into a service development, they will always be limited by two factors: their dependency on the particular technology used (WSDL in this case) and their lack of a robust theoretical foundation.

For these reasons In order to apply type theory constructs to service interfaces, assumes that each service description S is comprised of records s that represent the conceptual dependencies inside the service interface description. A service record s is a subtype of another record s0 if and only if it has at least all the typed properties of s0 (and possibly more) and all the common properties are also in a sub typing relation. In this case we write s _ s0.

IJCSMS International Journal of Computer Science & Management Studies, Vol. 12, Issue 02, April 2012
ISSN (Online): 2231 –5268
www.ijcsms.com

COSINE SIMILARITY

When documents are represented as term vectors, the similarity of two documents corresponds to the correlation between the vectors. This is quantified as the cosine of the angle between vectors, that is, the so-called cosine similar-ity. Cosine similarity is one of the most popular similarity measure applied to text documents, such as in numerous in-formation retrieval applications and clustering too. Each dimension represents a term with its weight in the document, which is non-negative. As a result, the cosine similarity is non-negative and bounded between [0,1].In other words, documents with the same com-position but different totals will be treated identically. Strictly speaking, this does not satisfy the second condition of a met-ric, because after all the combination of two copies is a differ-ent object from the original document. However, in practice, when the term vectors are normalized to a unit length such as 1, and in this case the representation of d and d 0 is the same.

URBE MATCHMAKER

The similarity algorithm running in URBE implements a similarity function fSim : (S; S0) ! [0::1] that, given two service descriptions S – representing the requested service, and S0 – representing the available service, returns the similarity degree as a value included in [0::1]: the higher the result of fSim, the higher the similarity between the two interfaces.

Fig 1 provides a high level view of fSim, in which each service can be represented as a three-level tree: first, we have S representing a portType, then the set of operations S:opk, and finally, the set of parameters (S:opk:inl, S:opk:outm) representing the parameters of the supported operations. As a consequence, the functions which evaluate the similarity among the whole interfaces (fSim), operations (opSim), and parameters (inP arSim and outP arSim) are nested in the same way. More specifically: fSim returns the similarity between S and S0.

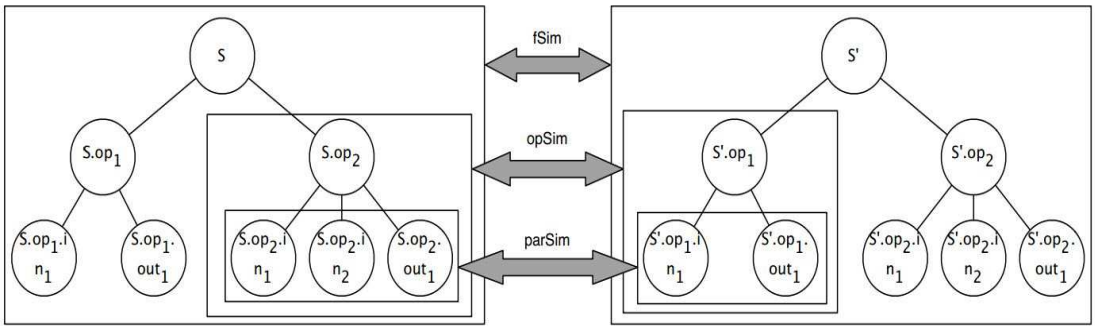

Figure 1 Tree-based representation and nested comparison in URBE

PROPOSED WORK

The practice of finding a suitable web service for a given application is termed as service discovery. For a particular function ample of services may be available. To find a service the suits the developer search is a major issue that has been addressed through solutions of diverse nature. Even if a matching service of users interest is found, but when it fails to synchronous with the application under Construction, then the process of discovery has to be redone with some change in the search

IJCSMS International Journal of Computer Science & Management Studies, Vol. 12, Issue 02, April 2012
ISSN (Online): 2231 –5268
www.ijcsms.com

string. Which make the problem more tedious? To solve the above, a measure called similar service discovery has gained importance, wherein similar services to the service in hand are searched for with the help of similarity measure algorithm, moreover a compatibility issue monitoring mechanism as a supplement to the solution improves the efficiency of finding a service a better service.

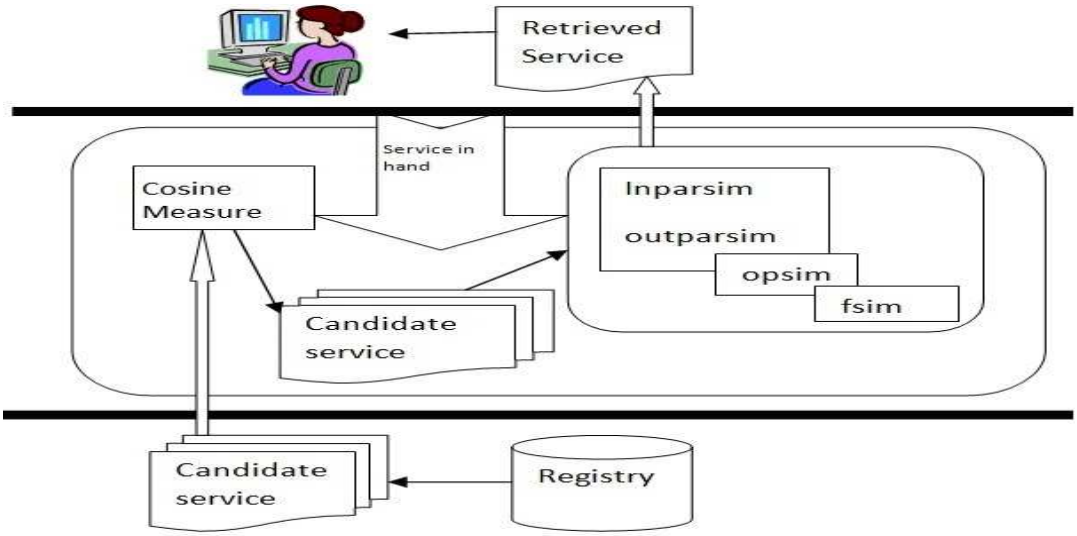

Fig 2: Proposed Architecture

The aim of the proposed system is to improve the efficiency as well as time consumed in the similarity measure. First the number of candidate service is reduced using conventional cosine measure which is used for matching documents.

Then the structure based similarity measure is used along with compatibility measure in the existing system.

CONCLUSION

It uses cosine measure to reduce search time. And also the QoS is also improved. Hence the most relevant and compatible service is retrieved. Despite of the above differences, these measures' overall performance is similar. Considering the type of cluster analysis involved in this study, which is partitioned and require a similarity or distance measure, we can see that there are three components that affect the final results representation of the objects, distance or similarity measures, and the clustering algorithm itself. This lead us to two directions for future work as follows. First, I plan to investigate the impact of using different document representation on clustering performance, and combine the different representations with similarity measures. In particular, the Wikipedia as background knowledge base, and enrich the document representation by adding related terms identified by the relationships between terms in Wikipedia. Wikipedia provides rich semantic relations between words and phrases, with a extensive coverage. This will also help to alleviate the problems with the bag of word document model, that words must occur literally and semantic relationships between words are neglected. Meanwhile, I plan to investigate the effectiveness of these similarity measures with a multiview clustering approach. In many cases we can view a given document from more than one perspective. For example, web pages intuitively provide at least

IJCSMS International Journal of Computer Science & Management Studies, Vol. 12, Issue 02, April 2012
ISSN (Online): 2231 –5268
www.ijcsms.com

three views—the content text appear in the web page itself, the anchor text of the outgoing links that are embedded in the page and the anchor texts from incoming links. Conventional clustering normally combines these different views (if they are taken into account at all) into a single mixed representation and uses it for clustering. We assume that by dividing the mixed representation up and using the different aspects individually can provide relevant information that is well separated according to its characteristics, therefore benefiting subsequent clustering.

REFERENCES:

[1]Vasilios Andrikopoulos and Pierluigi Plebani, "Retrieving Compatible Web Services" 2011 IEEE International Conference on Web Services, 10.1109/ICWS.2011.24

[2]G. Pirro´ and N. Seco, "Design, implementation and evaluation of a new semantic similarity metric combining features and intrinsic information content," in OTM '08: Proceedings of the OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008. Part II on On the Move to Meaningful Internet Systems. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 1271–1288.

[3]J. Farrel and H. Lausen, "Semantic annotations for WSDL and XML schema," http://www.w3.org/TR/sawsdl/, April 2007.

[4]R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval. ACM Press / Addison-Wesley, 1999

[5]R. Fang, L. Lam, L. Fong, D. Frank, C. Vignola, Y. Chen, and N. Du, "A version-aware approach for web service directory," in International Conference on Web Services (ICWS) 2007, Jul. 2007, pp. 406–413.

[6]R. Weinreich, T. Ziebermayr, and D. Draheim, "A versioning model for enterprise services," in Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on, vol. 2, 2007, pp. 570–575.

[7]V. Andrikopoulos, A Theory and Model for the Evolution of Software Services. Tilburg, Netherlands: Tilburg University Press, 2010, no. 262.

[8] S. R. Ponnekanti and A. Fox, "Interoperability among independently evolving web services," ser. Lecture Notes in Computer Science. Toronto, Canada: Springer Berlin / Heidelberg, 2004, pp. 331–351.

[9]P. Kaminski, M. Litoiu, and H. Muller,¨ "A design technique for evolving web services," in Proceedings of the 2006 conference of the Center for Advanced Studies on Collaborative research, ser. CASCON '06. New York, NY, USA: ACM, 2006.

[10]A. Brogi and R. Popescu, "Automated generation of BPEL adapters," in ICSOC 2006, ser. Lecture Notes in Computer Science. Springer, 2006, pp. 27–39.

[11] E Damiani, M. G. Fugini, and C. Bellettini, "A hierarchy-aware ap-proach to faceted classification of objected-oriented components," ACM Trans. Softw. Eng. Methodol., vol. 8, no. 3, pp. 215–262, 1999