IDENTIFYING CONTEXT SOURCES TOWARDS CONTEXT-AWARE ADAPTED WEB SERVICES

Georgia M. Kapitsaki
Department of Computer Science, University of Cyprus, 75 Kallipoleos Street, P.O. Box 20537, CY-1678, Nicosia, Cyprus

Keywords: Context, Context-awareness, Web services.

Abstract: Context-awareness refers to the ability of services and applications to proactively adapt their behavior to the characteristics of the execution environment, such as weather conditions, location, etc., namely context. Web services as the most popular implementation of service-oriented applications are usually exploited in this field. During the development process of such services an interesting challenge lies in identifying reusable context properties or services that constitute potential context sources for the adaptation to context. In this paper a solution for this context source matchmaking is proposed. The identification is performed by matching the WSDL specifications participating either as business services or as context sources. The procedure is demonstrated through an evaluation exploiting descriptions from online service registries.

1 INTRODUCTION

Context in service computing is used to describe different kind of information depicting specific conditions, such as weather temperature, location information, end-user preferences, current activity, etc. The most widely mentioned definition of context has been provided by Dey and Abowd (2000), whereas context-awareness refers to the ability of services to proactively adapt their behavior to contextual information. Additionally, in the latest years there is an increasing trend towards the provision of personalized and context-aware services to end-users both for web and mobile environments. Web Services (WSs) – being the most popular implementation of service-oriented applications – are usually exploited in such environments resulting to context-aware web services. This way web services can also act as reusable components for the creation of composite innovative applications consisting of individual web services. Web services can be exploited both as business components offering specific functionality to the composite application (e.g. hotel room reservation, stock options information retrieval) referred to as Business Web Services (BWSs), but also as context sources related to the retrieval of context information, referred to as Context Web Services (CWSs).

The process of combining these services at implementation level is an important issue that needs to be addressed by application developers. In this paper a process for the identification of potential context sources exposed as web services towards the development of context-aware adapted web services is proposed. In contrast to existing matchmaking works that concentrate on service selection, where a service request is matched against offered services, the current procedure revolves around context-awareness. The goal is to assist developers in identifying reusable matching services that are supported either by the provider intranet or by external service providers. The identification is based on the matchmaking of the corresponding service descriptions (BWSs and CWSs).

The rest of the paper is structured as follows: Section 2 discusses briefly the scientific background and related work, whereas Section 3 presents the principles of the matchmaking process. The process is. evaluated in Section 4 through various service descriptions. Finally, Section 6 concludes the paper.

2 RELATED WORK

In the framework of service discovery approaches towards service matching can be divided into end-user service discovery, which refers to runtime discovery prior to service execution, and developer service discovery, which aims at assisting developers in identifying potential service components. In the context of the current paper the focus is given on developer service discovery and more precisely on Web Service Description Language (WSDL) (W3C, 2001) based discovery that gives space for more flexible discovery approaches instead of catalogue browsing through Universal Description, Discovery and Integration (UDDI, 2004) registries, which can prove insufficient.

A significant approach towards service discovery for web services is provided by Wang and Stroulia (2003). It comprises of different stages resulting to the structural matching between WSDL specifications. The web service interface data types, messages, operations and web services are compared sequentially with different algorithms and the final results are ordered based on the overall computed matching score. A signature matching for WS search based on a full text analysis performed on WSDL files can be found in the work of Gannod and Bhatia (2004), whereas an event-driven rule-based context-aware system is presented by Yang, Zhang, and Chen (2008). The system is accompanied by a context-aware service oriented architecture that matches service requesters' attributes with the ones issued by the service providers. Works on WSDL comparisons can also be found in more recent approaches (Liu et al., 2010; Plebani and Pernici, 2009).

Context-awareness in WSs can be managed in different ways. Although the solutions are mainly more generic (pervasive computing applications, etc.), lately a variety of dedicated approaches focusing on the provision of context-aware web services have been proposed. In many cases the presented techniques rely on the interception of Simple Object Access Protocol (SOAP) (W3C, 2007) messages for injecting context dependent behavior (Keidl and Kemper, 2004; Prezerakos, Tselikas and Cortese, 2007). For more information on related work regarding context-awareness the reader can refer to the survey on context-aware service engineering presented by Kapitsaki, Prezerakos, Tselikas and Venieris (2009).

Having as motivation the above works this paper focuses on the development of an identification process for context-aware web services. For this matching base a different approach needs to be applied; the discovery techniques mentioned include matching between service descriptions and given keywords or between uniform service descriptions without capturing specific constraints (like context-awareness). Combinations of comparisons useful in revealing similarity concepts that are missing from the current literature are provided in the framework of this work. The presentation of the paper focuses on the applicability of the proposed scheme on real Web service descriptions. The main novelty of the proposed procedure lies in the use of matchmaking principles in the field of context-aware web services for context source discovery purposes.

3 MATCHING PRINCIPLES

3.1 Context Adaptation Cases

In previous work the research approach towards a context adaptation mechanism for web services through message interception has been presented (Kapitsaki, Kateros and Venieris, 2008). Following the rationale behind this aforementioned procedure three context adaptation cases for web services are assumed:

- **Parameter Injection:** this case refers to the possibility of having operations with input parameters that depend on context information. The respective request message parameters can be replaced with the actual context values. Examples include parameters expressing the name of a city or country, the end-user age, the outdoor temperature, etc.
- **Operation Selection:** when one service aggregates a number of methods offering the same functionality in different ways or using different parameters an operation selection adaptation may need to be performed. The operation to be invoked is selected based on context information (e.g. a payment service may select the method that corresponds to the payment choice stored in a user profile).
- **Response Manipulation:** it refers to the manipulation or the modification of service responses based on context and is often related with filtering or sorting operations (e.g. filter outdoor activities in bad weather conditions).

Combinations of the above are also possible. In the framework of an application consisting of web services different adaptation cases can also be met (e.g. service selection based on contextual parameters). However, the adaptation focuses on the above cases which are meaningful when the adaptation to context is assumed at the web service level. This choice guarantees also the preservation of the integrity of the exchanged messages between the service client and the web service.

3.2 Service Identification

The proposed context source matchmaking lies in the matching between web service descriptions and more specifically in the comparison of the elements contained in the WSDL descriptions of the business web services and the context web services. The similarity of the descriptors is calculated through appropriate scores that express the semantic distance of the different elements assigning higher values to concepts with closer proximity. The matching is performed for the first two context adaptation cases mentioned previously and is detailed in the next sections. The third case of response manipulation is neglected, since it is not possible to conduct an appropriate matching on this level – which permits many degrees of freedom – based on the information available in the web service specifications.

3.2.1 Case 1: Parameter Injection

For similarity related with parameter injection the input parameters of the BWS methods should match the return parameters of the CWS methods. The compliance needs to be maintained for both names and types of the participating parameters. At a second level the BWS input parameters need to be matched against the name of the CWS operation, which may also express the operation functionality or the meaning of the values returned (e.g. an operation named `getCityInformation(...)` will probably return information like the city name and postal code). The skeleton of the algorithm executed is shown below. `f_score` maintains the maximum value of the scores calculated for each operation.

```
foreach (BWS_operation in BWS_descr)
 foreach (BWS_input_param in
BWS_operation)
  foreach (CWS_operation in CWS_descr)
sc1=score(BWS_input_param.NAME,
CWS_operation.NAME);
   foreach (CWS_output_param in
CWS_operation)
sc2=max(sc2,score(BWS_input_param.
NAME+TYPE,CWS_output_param.NAME+TYPE));
   end
   f_score=max(score,sc1/2 + sc2/2);
  end
 end
end
```

3.2.2 Case 2: Operation Selection

In operation selection the name of the BWS operation is matched against both the CWS operation and the return parameter names. However, this comparison is meaningful for BWSs that aggregate two or more services implementing the same functionality in different ways (e.g., that would lead in choosing `getWinterActivities(...)` over `getSummerActivities(..)` operation in a tourist service, when retrieving activities in winter time). In order to check this similarity a test for the detection of similar business service methods in terms of method names and types returned is conducted first. The matching is then performed on the compatible operations of the business service (if any). The algorithm for this adaptation case is depicted below keeping the maximum score obtained in variable `f_score`.

```
foreach (BWS_matching_operation in
BWS_descr)
 foreach (CWS_operation in CWS_descr)
sc1=score(BWS_matching_operation.NAME
, CWS_operation.NAME);
  foreach (CWS_output_param in
CWS_operation)
sc2=max(s2,score(BWS_matching_operati
on.NAME, CWS_output_param.NAME));
  end
   f_score=max(score,sc1/2 + sc2/2);
 end
end
```

Based on the above cases the names of the pairs parameter–parameter and parameter–operation are compared, whereas parameters pairs are additionally compared for type similarity. Concerning type comparison a number of type groups for primitive (e.g. string, int, etc.) and complex data types (e.g. list, array, etc.) have been introduced. Nevertheless, the WSDL files may also contain special complex data types defined in the XML schema section that are exploited in the rest of the interface descriptor (e.g. types Wind or Movie). When performing the comparison, these types are broken down to their atomic elements, which participate in the type matching, unless the data type names and the corresponding namespaces coincide. If a specific structure for the input or return operation types is lacking, the keywords “Any” or “AnyType” are usually present in the WSDL specification.

The second type of comparison is related to the names of specific keywords, which acquires a higher weight in the overall score computation, since two semantically similar names are more important than the corresponding similar types. Non-meaningful words that do not add any semantic value to the service or operation meaning or that are irrelevant to context information properties (e.g. get, add, and, show, information) are neglected from the matching procedure. These non meaningful words are also indicated as stop words or words ”*too frequent to be* *meaningful*" (Falleri, Azmeh, Huchard and Tibermacine, 2010). A list of the 25 most common stop words in information retrieval is mentioned by Manning, Raghavan and Schütze (2008). A comparison based on the WordNet lexical database (WordNet) has been chosen for this step. Semantic lexical databases are an interesting field inspired from the general field of linguistics. WordNet is built around links expressing semantic associations among the contained concepts. Through WordNet different scores are assigned depending on the keyword sense proximity: original keyword, keyword synonyms, keyword hypernyms or keyword meronyms (4 for original keywords, 3, for direct synonyms and 1 for hypernym or meronym associations).

The matching through WordNet is first performed on the original names retrieved from the service operation and parameter names. If no match is found at this level, the procedure is repeated for all possible keyword substrings retrieved through a tokenization procedure. The tokenization is performed based either on capital letters (as noticed in the majority of words in WSDL descriptions) or on underscores (used in WordNet) identified in the respective names. Through the tokenization the keywords are broken down not only to the last level substrings, but also to the substrings containing more than one meaningful words. These substrings participate earlier in the matchmaking process. Indeed, it is preferable to have a composite substring match rather than a plain substring match, as it indicates closer similarity of the original input keywords. Therefore, the computed score is higher when the composite substrings from the two service descriptions match. For example, if the keywords `GetAirportInformationByCountry` and `CountryName` are to be matched, the matching will first be performed on the whole words, at a later stage on InformationByCountry and CountryName and even later on Country and Country.

4 EVALUATION PROCESS

An evaluation demonstrating the applicability of the proposed scheme on real web service descriptions has been conducted. For this purpose a number of descriptions obtained from online service registries have been exploited along with web services implemented in the framework of the current work. The BWSs for two adaptation categories are shown in Table 1: 1) adaptation based on the location of the requester, and 2) adaptation based on the point of time the request is made. Table 2 is dedicated to the presentation of the context web services depicting the category of context information provided: 1) Weather, 2) Location and 3) Time. The source of each service description is also shown in the tables.

Table 1: Business service examples.

Adaptation Category	Business Service
Location	Airport Information (WebServiceX)
	Driving (XMethods)
	Movie Information (XMethods)
	Proximity (XMethods)
Time	Foreign Exchange Rate (XMethods)
	Midnight Trader Financial News (XMethods)
	Historic Option data (XMethods)

Table 2: Context service examples.

Category	Context Service
Weather	CDYNE Weather (XMethods)
	Global Weather Service (XMethods)
	Global Weather (WebServiceX)
Location	IP2Geo (XMethods)
	IP2Location (XMethods)
	IPligenceGeoIPLocation (XMethods)
Time	DateTime (own implementation)

During the evaluation all available services have been matched in two ways: 1) through the described matchmaker that exploits the WordNet lexical database and 2) by performing plain syntactic matching on the description elements based on the introduced matchmaking cases. On keyword level the syntactic matcher performs a plain match on the keywords under examination instead of using WordNet (that can identify synonyms, meronyms, hypernyms and hyponyms). In order to provide a proof of concept for the results two widely adopted metrics are introduced from the information retrieval field: precision and recall. "*Precision is the proportion of retrieved documents that are relevant, and recall is the proportion of relevant documents that are retrieved*" (Voorhees, 1998). The metrics are defined as:

$$\Pr ecision = \frac{|\text{Relevant answers}|}{|\text{Total answers}|} = \frac{|Ra|}{|A|} \quad (1)$$

$$\operatorname{Re} call = \frac{|\text{Relevant answers}|}{|\text{Relevant documents}|} = \frac{|Ra|}{|R|} \quad (2)$$

Relevant answers refer to the compatible context service descriptions returned through the matching procedure, *total answers* is the whole service set returned (both compatible and incompatible descriptions) and *relevant documents* refers to the actual set of compatible context services. Values closest to one are indicators of more efficient information retrieval techniques. The results of the metric values for each BWS along with the names of the CWSs retrieved are summarized in Tables 3 and 4 for the semantic WordNet and the syntactic matching respectively. The services that appear in bold italics are the ones that actually match and are correctly returned through the procedure.

Table 3: Precision and Recall values for the WordNet matcher.

Request BWS and CWSs Returned	**Precision**	**Recall**
Airport Information (CDYNEWeather, ***IP2Geo, IP2Location, IPligenceGeoIPLocation,*** GlobalWeatherService, GlobalWeather)	0,5	1
driving (GlobalWeatherService, ***IPligenceGeoIPLocation***)	0,5	0,5
Foreign Exchange Rate (***DateTime,*** IPligenceGeoIPLocation, GlobalWeatherService, CDYNEWeather)	0,25	1
Historic option data (***DateTime***, CDYNEWeather IPligenceGeoIPLocation, GlobalWeatherService)	0,25	1
Midnight Trader FinancialNews (CDYNEWeather, GlobalWeatherService, ***DateTime***)	0,333	1
Movie Information (CDYNEWeather, ***IP2Geo, IP2Location, IPligenceGeoIPLocation,*** GlobalWeatherService)	0,6	1
Proximity (CDYNEWeather, GlobalWeatherService, GlobalWeather, ***IP2Location, IPligenceGeoIPLocation***)	0,4	0,667

Generally, the values obtained for the precision and recall metrics indicate that the proposed scheme can prove useful for software engineers towards the automatic identification of reusable context sources. The recall calculation using the WordNet matcher appears to have a slightly better performance than the plain syntactic matching. This is mainly observed, since the semantic similarities captured between the different keywords by the WordNet matcher include all context adaptation cases for the specific web service. Nevertheless, this is not the case with the precision values, where the syntactic matching has a better performance in some cases. This appears mainly because of the presence of keywords in the WSDL descriptions not directly related to the main service functionality. These keywords are, however, captured by the WordNet matcher, because of the semantic similarities they possess. This can be noticed for example when some weather services are returned together with the location adaptation case, since they provide weather forecasts for specific cities or countries, which have close meaning to location.

Table 4: Precision and Recall values for the syntactic matcher.

Request BWS and CWSs Returned	**Precision**	**Recall**
Airport Information (CDYNEWeather, ***IP2Geo, IP2Location, IPligenceGeoIPLocation,*** GlobalWeatherService)	0,6	1
driving (GlobalWeatherService, ***IPligenceGeoIPLocation***)	0,5	0,5
Foreign Exchange Rate (***DateTime,*** IPligenceGeoIPLocation, GlobalWeatherService, CDYNEWeather)	0,25	1
Historic option data (***DateTime***, CDYNEWeather IPligenceGeoIPLocation)	0,333	1
Midnight Trader FinancialNews (GlobalWeatherService)	0	0
Movie Information (CDYNEWeather, ***IP2Geo, IP2Location IPligenceGeoIPLocation***, GlobalWeatherService)	0,6	1
Proximity (GlobalWeatherService)	0	0

A general disadvantage identified in WordNet is that it is not possible to determine whether a given keyword is a verb or an adverb or another part of speech driving this way to wrong results (e.g. this problem appears with the keyword "point", which may refer to a point in time or a point in space). This drawback has also been identified for earlier versions of WordNet in earlier works (Mandala, Takenobu and Hozumi, 1998).The analysis of the given operation and parameter names as phrases that consist of nouns and verbs in the right order as specified by linguistics studies may improve the performance of the procedure.

5 CONCLUSIONS

In this paper the initial work towards a solution for the identification of potential context sources serving the development of context-aware web services has been proposed. The results are quite interesting and illustrate the value of the proposed procedure that alleviates the developer's work by allowing the reuse of available and already tested software components. The procedure can parse, analyze, normalize and compare service descriptions successfully for the majority of cases.

Future directions of the current procedure include the extension of the validation and evaluation during the whole development lifecycle, i.e. business services should be matched against appropriate context information at different abstraction levels, which is meaningful especially in the framework of engineering techniques applied with success in the latest years, such as Model-Driven Engineering (MDE) (Schmidt, 2006). Ongoing work includes also the introduction of non-functional properties extracted from the service documentation (e.g. communications protocol restrictions, etc.) towards the implementation of a service repository that captures the various service description aspects. This way more accurate results can be achieved in the matching procedure.

REFERENCES

Dey, K., Abowd, G. D. (2000) 'Towards a Better Understanding of Context and Context-Awareness'. In *CHI'00, Workshop on The What, Who, Where, When, and How of Context-Awareness, as part of the 2000 Conference on Human Factors in Computing Systems*. ACM Press.

Falleri, J.-R., Azmeh, Z., Huchard, M. and Tibermacine, C. (2010) 'Automatic *Tag Identification in Web Service Descriptions'. In WEBIST'10, International Conference on Web Information Systems and Technology*.

Gannod, G. C. and Bhatia, S. (2004) 'Facilitating Automated Search for Web Services'. In *Proceedings of the IEEE International Conference on Web Services*. IEEE Computer Society Press, pp. 761-764.

Kapitsaki, G. M., Kateros D. A. and Venieris, I. S. (2008) 'Architecture for Provision of Context-aware Web Applications based on Web Services'. In *PIMRC 2008, IEEE Personal, Indoor and Mobile Radio Communications*. IEEE Computer Society Press, pp. 1-5.

Kapitsaki, G. M., Prezerakos, G. N., Tselikas N. D. and Venieris, I. S. (2009) 'Context-aware Service Engineering: A Survey'. *Journal of Systems and Software*, vol. 82, no. 8, pp. 1285–1297.

Keidl, M., and Kemper, A. (2004) 'Towards Context-Aware Adaptable Web Services'. In *WWW'04, 13th international World Wide Web conference*. ACM Press, pp. 55-65.

Liu, F., Shi, Y., Yu, J., Wang, T. and Wu, J. (2010) 'Measuring Similarity of Web Services Based on WSDL'. In *2010 IEEE International Conference on Web Services*, pp. 155-162.

Mandala R., Takenobu T. and Hozumi T. (1998) 'The Use of WordNet in Information Retrieval'. In *COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, pp. 31-37.

Manning, C. D., Raghavan, P. and Schütze, H. (2008). '*Introduction to Information Retrieval'*, Cambridge University Press.

Plebani, P. and Pernici, B. (2009) 'URBE: Web Service Retrieval Based on Similarity Evaluation'. *IEEE Transactions on Knowledge and Data Engineering*, 21 (11), p. 1629-1642.

Prezerakos, G. N., Tselikas, N. and Cortese, G. (2007) 'Model-driven Composition of Context-aware Web Services Using ContextUML and Aspects'. In *ICWS'07, IEEE International Conference on Web Services*, IEEE Computer Society Press, pp. 320-329.

Schmidt, D. C. (2006) 'Model-Driven Engineering'. *IEEE Computing*, vol. 39, no. 2, Cover Feature, pp. 25-31.

UDDI Specification, 3.0.2, (2004), http://uddi.org/pubs/uddi-v3.0.2-20041019.htm.

Voorhees, E. M. (1998) 'Using WordNet for text retrieval. In WordNet, An Electronic Lexical Database', The MIT Press, pp. 285-303.

Wang, Y. and Stroulia, E. (2003) 'Flexible interface matching for Web-service discovery'. In *WISE 2003, 4th International Conference on Web Information Systems Engineering*. IEEE Computer Society Press, pp. 147-156.

WordNet. http://wordnet.princeton.edu/.

W3C. (2007) 'Simple Object Access Protocol (SOAP), 1.2', http://www.w3.org/TR/2007/REC-soap12-part0-20070427/.

W3C. (2001) 'Web Services Description Language (WSDL), 1.1', http://www.w3.org/TR/wsdl.

Yang, S., Zhang, J. and Chen, I. (2008) 'A JESS-enabled context elicitation system for providing context-aware Web services'. *Expert Systems and Applications*, vol. 34, issue 4, pp. 2254-2266.