ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)
On 10th & 11th February Organized by

Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

A DSL Query Model Language for Contextual Discovery of Services

Bhaswini D[1], J Mathi Sankari[2]
P.G Student, M.E CSE, Dhaanish Ahmed College of Engineering, Chennai, T.N, India. [1]
Assistant professor, Dept. of M.E CSE, Dhaanish College of Engineering, Chennai, T.N, India. [2]

Abstract— A DSL is a query language specialized to a particular application domain. Creating a domain-specific language (with software to support it), rather than reusing an existing language. It can be worthwhile if the language allows a particular type of problem or solution to be expressed more clearly than an existing language would allow and the type of problem in question reappears sufficiently often. A search engine is used by software developers to find an appropriate web services for their application where he may not know whether a services already exists or not. This may lead to reinvent the same concept again which will affect production phase of a company. To avoid such an issue, we have developed a new query language DSL which will be very much easy for a software developer to communicate. We have defined a repository service to extract the contextual and Meta information from the service definition and store in the repository. This will avoid affecting the results because of improper queries. Generally service is a restful service which takes XML/JSON as input and gives responses in XML/JSON based on HTTP content negotiation from client.

Keywords- Contextual Information, DSL Language, Web Services Interoperability.

I. INTRODUCTION

Several new and innovative front end applications have been developed for performing for mining web services. These include several GUI tools that help to analyze, and query databases and spreadsheets as a means to gather information for organizational decision making. A generic query language is used for heterogeneous web service to access contextual information. In most cases, queries can only express search criteria that constrain trivial, service level properties such as the service name, provider, or classification, while meaningful requirements on more important properties of the desired service operation cannot be directly expressed. As a result, the query response must often be further processed by the service requester, to check if the included service operations suit their needs. Since the query evaluation mechanisms are mostly limited to keywords-based matchmaking, which yields poor results in terms of precision and recall, such processing becomes a considerably cumbersome procedure.

The generic query model is used for domain specific language for accessing all contextual information. It will be more flexible and efficient to compare with existing query model. The abstractness and the expressiveness of DSLs, they can easily read programs, and learn the languages in order to specify the programs with efficiency, or in another words, with little time spent.

Another fact of that efficiency can be observed on the tools that give support to the language. The system defines a repository for Extracting the Meta and contextual information for the service defined in an organization and stores in the registry. A Metadata repository is a database created to gather, store, and distribute contextual information about business data, when documented it is known as metadata. This contextual information of business data include meaning and content, policies that govern, technical attributes, specifications that transform, and programs that manipulate.

DSL method of querying will help to match the Meta information available in web services. It is based on JSON to find out queries. DSL is created specially to solve problems in a particular domain. To discover Web services, a service registry is needed. This requires describing and registering the Web service. Publication of a service requires proper description of a Web service in terms of business, service, and technical information.

II. RELATED WORK

In this section, we evaluate and discuss the related work focusing on the proposed query models, Many service discovery frameworks such as OWL- S and WSMO have embraced the Semantic Web technologies in order to improve the matchmaking process. Among the first attempts to build upon such frame- works, the OWL-S/UDDI matchmaker is based on the OWL-S language for semantically describing Web services, and provides extensions to the UDDI query model to accommodate the evaluation of semantically enhanced queries. Similarly, the OWL-S language has been applied to P2P service discovery in JXTA networks, through ODEN. Being tightly coupled with the respective technologies and their conceptual models, these approaches are not applicable to the discovery of other types of services, while queries can only ex- press requirements towards the closed set of service properties that are supported by OWL-S, namely the inputs, outputs, preconditions, and effects of the desired Web service. It should be noted though that, despite these limitations, these approaches could accommodate additional requirements, thanks to the extensibility of the UDDI and JXTA protocols, respectively. In any case, unlike Proteus, users are not supported in prioritizing their requirements, while matchmaking is performed on the basis of four predefined degrees of match, which might leave partial, yet potentially useful matches out of the query response.

In conclusion, compared to the investigated query models, we argue that our approach is more appro- priate for the discovery of heterogeneous services, as it provides a unified interface to facilitate the work of service requesters operating in different environments.

In chapter three we discuss about the proposed method. Chapter four describes the implementation. Chapter five describes the attacks and countermeasures. In chapter six describes the experimental results. In Chapter seven the conclusion of the paper and suggests for the future improvements of the system.

III. PROPOSED SYSTEM

In the proposed system, the system defines a repository for storing the Meta and contextual information for the service defined in an organization. Introducing a new service property and its corresponding requirement in the model does not affect its structure or its evaluation mechanism The system will also have a new Domain Specific Query Language for searching the contextual service information from the repository. This DSL query language will be defined by developers/IT managers who want to search the enterprise repository to get contextual service information. The Search web application will be developed using spring. The repository itself will be JAX-RS web service.

1. Service Registration: It is the process of specification of web services to the system. New services are registered in registry through service registration process. There are several registries that different companies (service providers) maintain and all of them are synchronized after regular interval.
2. Service Request: Clients that need a particular service send request through service request module.
3. Translator: The purpose of translator is to translate request/response from one form to another. We use translator so that all of the services are registered from external form to a form used by system and vice versa.
4. Web Server: Registries are hosted on web server on World Wide Web. Services exchange data directly with various databases and web servers, that implements decentralized data flow.
5. Evaluator: The evaluator evaluates selected web services on the basis of interface based and functionality based rules and returns the best selected service based on specified criteria.
6. Web: In proposed framework, web is World Wide Web network where all service providers register their web services in UDDI registries. If desired web services are not found in repository or database then matching engine will search them from UDDI registries and save it in database for current and future use.
7. Composer: The composer composes the selected component services in order to make a single desired web service.
8. Matching Engine: The purpose of matching engine is to match the user's request from the web services database. If match is found, it returns results back to web server. If not then select the web services from web, store/update them in database and then return results back to requested composer.
9. Service Registry: The service registries are used to register the web services by web service providers. Also they are used to request the user's desired web services. Each registry has the references of web services that are actually hosted on service repositories.

IV. PROBLEMS

The main interest of web service compositions is to give value-added services to existing web services and introduce automated web services. Also they provide flexibility and agility. There are few problems in dynamic web service composition.

• First, the number of web services is increasing with time and it is difficult to search the whole repository for desired service in order to use it for the fulfillment of specific goal.

• Second, web services are dynamically created and updated so the decision should be taken at execution time and based on recent information.

• Third, different web service providers use different conceptual models and there is a need of one structure so that web services easily access each other without any technical effort.

• Forth, only authorized persons can access few of these web services.

V. IMPLEMENTATION

JAVA programming language is used to implement the proposed technique. We have used Apache JUDDI (Java Implementation of Universal Description Discovery and Integration version 3), which is java implementation of UDDI Registries. RUDDI is used to access the JUDDI. Service providers can perform various operations in UDDI registries like save, edit and delete services and businesses by using RUDDI. JAXR (Java API for XML Registries) and UDDI4J also provides the java API to access UDDI registries but the drawback is that JAXR is Java API to access various types of XML registries like UDDI, XML and others. It is not specific to UDDI registry. Whereas UDDI4J supports only UDDI version 2. We have also used WSDL4J (Web Services Description Language for Java Toolkit) which is Java API used to access WSDL files. WSIF (web services invocation framework) is used for invocation of multiple services from different repositories. Service providers register their web
services in UDDI registries. Requesters requests for web service and translator translates it. Then matching engine search the requested service from web services database. The
valid services are returned to evaluator and send the evaluated
services to composer. Composer sends the composed service to execution engine and returns the resultant service to requester through translator. If web services are not found in WSDB then matching engine will search from UDDI registries. The matched services are returned to evaluator which evaluates them and after composition resultant service is passed to execution engine. Execution engine execute these
services and results are sent to requestor through translator.

The implementation of DSL can be carried out using several approaches:
Interpretation or compilation This is the classical ap- proach to implementing a new language. Standard compiler tools , can be used, or tools dedicated to the implementation of DSLs like Draco, ASF+SDF, Kephera ,Kodiyak ,design by selection, or InfoWiz .The main advantage of building a compiler or interpreter is that the implementation is completely tailored towards the DSL and no concessions are necessary regarding notation, primitives and the like. Also, error detection, static analy- sis, and optimizations can be done at the domain level, for example using an effect system as in. Clearly, an important problem is the cost of building such a compiler or interpreter from scratch, and the lack of reuse from other (DSL) implementations, although some DSL tool sets (for example InfoWiz) are particularly designed to overcome such problems.
As an alternative to implementing a DSL from scratch, a DSL can be implemented by extending a given base lan- guage. For instance, describes an extension of (a restricted version of) a general-purpose language with domain-specific constructs. The main advantage of this approach is that all features of the base language remain available and need not be re-implemented. When implementing domain-specific extensions of a base language, the implementation of the base language can be reused in three different ways:
Embedded languages / domain-specific libraries In this approach, existing mechanisms such as definitions for func- tions or operators with user-defined syntax are used to build a library of domain-specific operations. The syntactic mech- anisms of the base language are used to express the idiom of the domain. An advantage of this approach is that the compiler or in- terpreter of the base language is reused as is for the DSL. The main limitation is in the expressiveness of the syntactic mechanisms in the base language. In many cases, the op- timal domain-specific notation has to be compromised to fit the limitations of the base language. Typical examples of this approach are (a robot control language embedded in Haskell) and (a PIC-like drawing language embedded in ML). The concept of domain-specific embedded language was coined by Hudak .
Preprocessing or macro processing : In this approach the new constructs are translated to statements in the base lan- guageby a preprocessor. The main advantageof this approach is simplicity. Its main disadvantage is that static checking and optimization are not done at the domain level. Consequently, generated code is error prone, and the user is provided with feedback on these errors at the level of the base language, or only at run- time.
Extensible compiler or interpreter This approach is sim- ilar to the previous one, but the preprocessing phase is now integrated in the compiler. The advantage is that more type checking and better optimization is possible. The Tcl interpreter is also a prime example: it has been extended for dozens of domains.

ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)
On 10th & 11th February Organized by

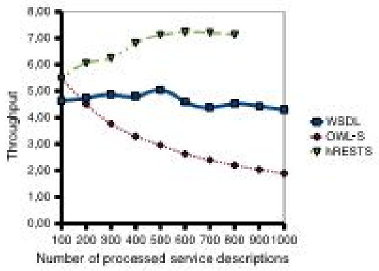
Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

Apart from building a dedicated DSL compiler or interpreter, or reusing the implementation of an underlying base language, other implementation techniques may be used. For instance, in aspect-oriented programming a DSL is used to describe an aspect of a system's behavior that is orthogo- nal to its main functionality. An aspect weaver is then used to generate domain-specific code and merge it with the main code.

VI. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

A. Experimental Results

An experimental evaluation of our approach, so as to assess its efficiency, in terms of the accomplished recall precision of the query results, while also investigating its feasibility, by measuring the various computational costs. Within our goals was also to verify that, Proteus is capable in operating in a variety of service settings. For the purposes of the evaluation, we implemented a prototype of the Proteus engine in JavaTM 6. All measurements were taken by deploying and executing the engine on a 2.4 GHz Intel Core 2 Duo processor, with 2 GB of RAM, running Mac OS X 10.5, whereas a Java HotspotTM 64-bit Server VM was used by the Java Runtime Environment (JRE). The experiments were conducted in two separate phases. The first phase involved the preparation of an adequate set of heterogeneous environments by means of the DSL Crawler, while the second one involved the generation and execution of the queries by means of the DSL Query Processor.



B. Performance Evaluation and Analysis

In the proposed technique, UDDI registries provide efficient access. The efficiency is further improved since composite services once discovered, are stored in local WSDBs and till the timer expires these composite service can be used locally. Replicated WSDBs make the process reliable. Associated timestamp in WSDBs makes it possible to use any new services that become available on the registry. The purpose of our algorithm is to invoke and compose web services efficiently. The effectiveness of algorithm is evaluated by calculating the method exposed time and service composition time. We have taken different number of methods exposed and calculated their respective time. We have also taken different number of composed services and calculated their respective execution time. It is clearly shown in Figure 1. Figure 2 shows the different number of methods exposition time.

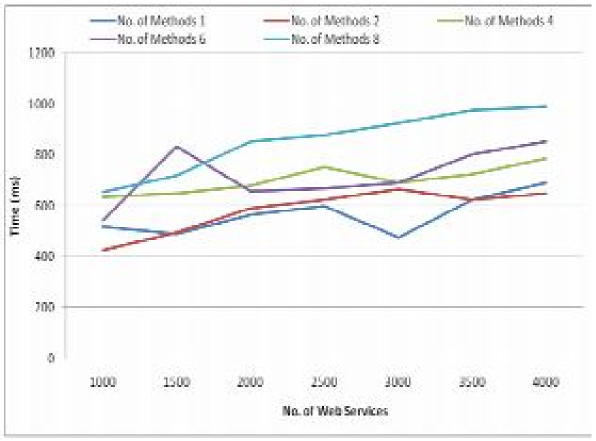


Fig 5: Efficiency of proposed algorithm

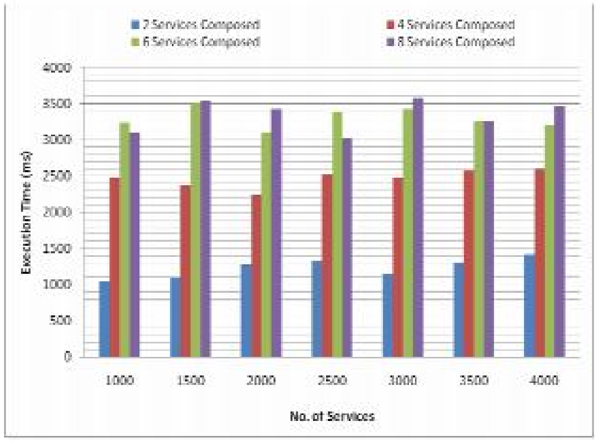


Fig 6: Efficiency of proposed algorithm

VII. CONCLUSION

In this paper, a common back-end can be used to compose high performance, compiled domain-specific languages. The common back-end also provides a means to achieve meaningful reuse in the compiler implementations when targeting heterogeneous

ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)
On 10th & 11th February Organized by

Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

devices.This system will help developers to find the exact web services. If they want to use the criteria, they have to specify and our system will make the most suitable services provided currently by searching the web services web data. This will make the developers work easy in the sense that they will not reinvent the wheel again. Our model will overcome the existing problems by providing proper search results thus makes the searching process simple and time reducing which will boost the company production phase. Thus we are creating registry to extract and store the Meta information.

REFERENCES

[1] Sellami,S.; Boucelma,O."Web Services Discovery and Composition: A Schema Matching Approach",Web Services(ICWS), 2011 IEEE International Conference on Digital Object Identifier:10.1109/ICWS.2011.105,Publication Year:2011,Page(s):706

[2] FangfangLiu ; LeiWang ; JieYu, "Context aware similarity search of web service " Information Science and Technology (ICIST) , 2012 International Conference on Digital Object Identifier: 10.1109/ICIST.2012.6221715, Publication Year: 2012 , Page(s): 596

[3] Kumara,B.T.G.S.; IncheonPaik;GyeongmuLee "Ontology learning ",Computing and Convergence Technology (ICCCT), 2012 7th International Conference on Publication Year: 2012 , Page(s): 705 – 710

[4] Skoutas,D.; Sacharidis,D.; Simitsis,A.; Sellis,"Ranking and Clustering Web Services Using Multicriteria Dominance Relationships " T.Services Computing, IEEE Transactions on Volume: 3, Issue: 3Digital Object Identifier: 10.1109/TSC.2010.14, Publication Year: 2010, Page(s): 163 - 177 Cited by: Papers (13).

[5] Bin Xu; Sen Luo; Kewu Sun,"Towards Multimodal Query in Web Service Search"
Web Services (ICWS), 2012 IEEE 19th International Conference on Digital Object Identifier: 10.1109/ICWS.2012.42 Publication Year: 2012 , Page(s): 272 – 279

[6] D.Elenius and M.Ingmarsson, "Ontology-Based Service Discovery in P2P Networks," Proc. First Int'l Workshop Peer-to-Peer Knowledge Management (P2PKM '04), 2004.

[7] M. Klusch, B. Fries, and K. Sycara, "Automated Semantic Web Service Discovery with OWLS-MX," Proc. Fifth Int'l Joint Conf. Autonomous Agents and Multiagent Systems (AAMAS '06), 2006.

[8] M.Klusch and F.Kaufer, "WSMO-MX: A Hybrid Semantic Web Service Matchmaker," Web Intelligence and Agent Systems, vol. 7, no. 1, 2009.

[9] U.Lampe, S.Schulte, M.Siebenhaar, D.Schuller, and R.Steinmetz, "Adaptive Matchmaking for Restful Services Based on hRESTS and MicroWSMO," Proc. Fifth Int'l Workshop Enhanced Web Service Technologies, pp. 10-17, 2010.

[10] S. Kona, A. Bansal, L. Simon, A. Mallya, G. Gupta, and T.D. Hite, "USDL: A Service-Semantics Description Language for Automatic Service Discovery and Composition," Int'l J. Web Services Research.