A Novel Approach for Web Service Parameters Validation Using Ontology

This Sample Work has been completed by 'Tutors India'

www.tutorsindia.com

Table of Contents

ABSTRACT 3
1. INTRODUCTION 3
2. PREVIOUS WORK 4
A. Definitions of Concepts 4
B. Semantic Description Languages of Web 5
3. EXISITING SYSTEM 6
4. PROPOSED SYSTEM 7
4.1 Semantic Annotation Framework: Architecture 7
4.1.1 Ontology Based Service Index Annotator 7
4.2. Ontology construction for online shopping and extending wsdl 8
4.3. Annotation Verification Process 8
5. Test Adequacy Criteria 12
6.0 UML Diagrams 12
6.1. Class Diagram 12
6.2. Use Case Diagram: 12
6.3. Sequence Diagram 13
7. RESULTS 13
8. CONCLUSION 13
References 14

ABSTRACT

Semantic annotations of web services are useful in service discovery only when it reflects accurate service. But the rate of defects in semantic annotations was found to be very high due to the need for human interaction and expertise. The purpose of this work is to verify the semantic annotation of web services. The verification is based on the conventional testing process. In this paper, emphasis is laid on automation of semantic annotations to increase the efficiency and accuracy to provide useful web service through which the required data can be reproduced according to the specification defined by the services. In particular, the study proposed a technique from software testing to address the problem of semantic annotation verification. A software testing has been well proven to be effective for verification of software program behaviour according to its specifications. By using a test suite which consists of test cases with specific data values for feeding the execution, therefore the expected outputs are generated as per the specification. As specified by the test cases, the software is executed using the input values. In this way, there is a possibility to identify a defect due to the inconsistency between the output delivered and expected by a test case. The advantages of proposed model are: (1) ease of use when information, (2) automatic compliance to system requirements and specification,(3) improved accuracy (4) use of ontology based partitioning (5) Efficient defect detection mechanisms.

1. INTRODUCTION

The rapid development of the Internet in recent years had resulted in the emergence of large number of web services. The Web is now evolving into a distributed device of computation from a collection of information resources such as to find the weather, travel schedules, and the best restaurant while travelling. In addition, Web services also been used as the technology for business process execution and application integration. Although Web services are based on widely accepted standards, the lack of a formal description of the meaning of their functionality and the data exchanged has been a significant roadblock in the realization of integration. As the number of Web services increase, it is important to discover and compose Web services. This task forces the computer to do more of the necessary work where the Semantic web (www.w3.org/2001/sw/) addresses this concern [1] and expected to become a next-generation of the web. Semantic Web, aims at developing a global environment on top of web with interoperable heterogeneous organizations, humans, data repositories, web services, agents and applications.

Semantic Web Services offer the possibility of highly flexible web service architectures, where new services can be quickly discovered, orchestrated and composed into workflows [2] The extent of description available in the current WSDL standard leaves room for ambiguous interpretations of the functionality and data of a Web service. To address these issues semantic annotation of Web service elements are proposed, which explicates the exact semantics of the Web service data and functionality elements that are crucial towards the use of the Web service. This is done by annotating the Web service elements with concepts in domain models or ontologies. Since ontologies represent an agreed upon view of the modeled domain, any ambiguity in the interpretation of functionality or data of a Web

service is eliminated. The purpose of annotating Web services is to enable unambiguous and automated service discovery and composition.

Yet, semantic annotations of web services are useful in service discovery only when it reflects accurate service [3]–[5]. But the rate of defects in semantic annotations is found to be very high due to the need for human interaction and expertise. This is because, annotator neither involved in the domain ontology construction and web service development. Presently there are no efficient mechanisms to check the high rate of defects occurring while using semantic annotations. This makes the system prone to disuse and is found to be highly ineffective. Since web service providers supply annotations, therefore one would expect significantly low errors. Yet, the integration of annotation failed to integrate with lifecycle development of web services which resulted in annotation of third parties. In a typical scenario, an annotator is required to analyze the defects and he/she should be involved in the construction of ontology and in the development of web service, therefore there is a possibility for mistaken interpretation of behaviors and concepts while selection of annotation.

The structure of the paper is as follows: Section I critically reviews the previous studies conducted elsewhere on semantic annotations. This section is followed by brief description of existing system (Section 2) and subsequently the proposed system (Section 3). Section 4 presents the verification of semantic web service annotation using conventional software testing technique based on the domain ontology. This section also presents the verification process that includes multiple phases. The test cases used in semantic annotation is used to reveal the defects in the web service. The algorithm used in the testing technique reduces the effort of the manual annotator. The execution of algorithm results in normal or abnormal termination, which in turn determines the valid or invalid input. The most outstanding contribution of this work is the foundation of a novel technique, which would cover both the ontology concepts applied for annotation and its effectiveness of the adequacy criteria by applying that in real-world situation (section 5), in the case of online shopping service.

2. PREVIOUS WORK

Recently, Web services related research area are found to be active and the following section reviews the previous major techniques, platforms, standards that have been applied for web services closely related to present research.

A. Definitions of Concepts

The definitions of concepts used in this study are described as follows[6]: Ontology: This term is understood from a philosophical point of view where it is defined as “the knowledge of what is to be in oneself”. In the case, data processing it refers to the structured set of knowledge in a domain. It is an explicit share specification in a specific domain. Semantic annotation: A semantic annotation is referent to a relative ontology where it is a link to its semantic description assigned to an entity in the text. Reasoner: Based on the specific request it matches service advertisement such as input and outputs.

Matching: Based on some similarity features, the matching operates two concepts such as inputs, output and precondition and effects. This was done as per the Paolocci et al. [7]

B. Semantic Description Languages of Web

Several web service frameworks are promoted for W3C standardization where the most frameworks are evolved from the WSDL-S Specifications. The prominent ones are OWL-S (Ontology Web Language for Services), WSMO (Web service Modeling Language), WSDL-S (Web service Description Language Semantic) and SAWSDL (Semantic Annotation for Web service Description Language). These frameworks are reviewed based on the criteria such as Resources, property, language, annotation, model and matching.

Various studies have been conducted previously and the following are the studies that are closely related to our concept. Semnatic web access control policies were discussed by Bonatti et al.[8], while Denker et al.[9] developed the security annotations. Ankolekar et al. [10] proposed DAML-S and DAML + OIL ontology to explain the web services capabilities and properties. Further, author also described three aspects of ontology that includes the service grounding, the process model and the service profile, where later provides the information required for an agent to service discovery while service model provides information for an agent to make use of a service. In specifically, author has applied service grounding which connects with low level XML based description of web services. The research although shed light on upper ontology but still it requires human expert for service description, time consuming and does not guarantee for successful deployment. Medjahed et al. [11] proposed an ontology based framework for the automatic composition of web services developed from high-level declarative descriptions and implemented in e-government application. Author in this study defined formal safeguard for meaningful composition through the use of composability rules which compares the semantic and syntactic features of web services. Although the service composition is based on the ontology and syntactic and semantic riles, while composite service based on high level declarative descriptions still it failed to address the spatial and temporal availability. Further semantic features did not address before deployment of web service and composite services are not based on composition quality.

Lord et al. [2] proposed light-weight semantic discovery architecture and attempted to fits in the larger architecture of the Grid project, specifically for user requirements of bioinformatics. Further this paper provides light weight semantic support and usage of RDF's as backend provides low complexity. Yet, still system required experts in the domain ontology and the proposed failed to provide verification in the annotations therefore, susceptible to be applied to different domains. Patil et al. [12] proposed a Meteor-S web service annotation framework for semi-automatically marking up web service descriptions with ontologies. Author in this study has proposed algorithms to match and annotate WSDL. Although semantic annotations are made with real ontologies with higher quality still the paper failed to provide verifications to check the accuracy of annotation before deployment.

Gomadam et al. [13] presented a faceted approach to search and rank web APIs considering attributes or facets of the APIs as found in their HTML descriptions based on their utility and popularity. Although

the proposed provides effective search and classification mechanism for Web API's based on the serviut (service utilization ranking), still the proposed system assists human users for annotating web services which demands high costs at the same times there is a possibility of high rate of false positive. Maximilien and Singh [14] addressed the issue related to Quality of Service (QoS) via an agent framework coupled with QoS ontology. The proposed system supported multi-attribute utility and dynamic descriptions of web services and therefore QoS support. However, there is a possibility that proposed system may be vulnerable for attacks such as proofing while agencies data by malfeasant agents. This reflects that there is a lack of guarantee for accurate reflection of web service in semantic annotation. Davidson and Freire [15] applied provenance in scientific workflows to allow for result reproducibility, sharing, and knowledge re-use in the scientific community. Jiang and Luo [16] proposed extended UDDI which takes into account the height factor of ontology tree and local density factor. The findings revealed that the proposed service enhanced the web service discovery efficiency. However, this algorithm ignored quality of services, precondition and effects of web service operations. Boukhadra et al. [17] proposed semantic interoperability in a heterogeneous, distributed architecture based on the automatic dialing services semantic web. Saquicela et al. [18] proposed lightweight semantic annotation of RESTful services using cross-domain ontology and findings showed satisfactory results. Further, several tools like APIHUT, KINO [19], Meteor-S [20], APIHUT [13]are some of the tools that have been proposed to enhance the verification of semantic annotations of web services.

The proposed system combines provenance in workflow system with higher quality ontologies. Yet, the proposed model need an efficient way for managing data and failed to combine provenance workflow with database and be visualized.

3. EXISITING SYSTEM

The existing semantic annotation, described web service parameters partially but do not attempt to describe its transformation, which is critical in web service discovery. In addition, techniques proposed earlier on semantic annotations did showed following drawbacks such as a) manual annotation needs experts in domain ontology b) Manual annotation is time consuming task c) Semantic annotations are created for describing web services under strong assumptions d) Defect detecting power is very low and e) Test case generation and execution is expensive [2], [11], [13], [14], [21]. There are dearth of tools and techniques to verify the accuracy of annotation. Further, several studies[22], [23] have been conducted earlier to improve the Quality of Service provided to the end user. The objective is to enhance user experience and enable dynamic detection of requirements and avoid rigidity in creating and reusing information. In almost every system, accuracy seems to have emerged as a very serious threat to its effective implementation. Ontologies enable Web services framework to be machine readable, and this in turn allow software agents to handle more the decision making in completing a task [24], thereby creates effective use to annotations technology [25].The semantic annotations associate with relevant data structure in the ontology thereby evolving effective defect detection systems[26].This enables effective use of annotations technology to extract relevant information as per a dynamic specification and service requirements.

4. PROPOSED SYSTEM

In this paper, emphasis is laid on automation of semantic annotations to increase the efficiency and accuracy to provide useful web service through which the required data can be reproduced according to the specification defined by the services. In particular, the study proposed a technique from software testing to address the problem of semantic annotation verification. A software testing has been well proven to be effective for verification of software program behaviour according to its specifications [27]. By using a test suite which consists of test cases with specific data values for feeding the execution, therefore the expected outputs are generated as per the specification. As specified by the test cases, the software is executed using the input values. In this way, there is a possibility to identify a defect due to the inconsistency between the output delivered and expected by a test case. The advantages of proposed model are: (1) ease of use when information, (2) automatic compliance to system requirements and specification, (3) improved accuracy (4) use of ontology based partitioning (5) Efficient defect detection mechanisms.

4.1 Semantic Annotation Framework: Architecture

4.1.1 Ontology Based Service Index Annotator

In this paper, we proposed a semi-automatic semantic annotation framework based on the existing semantic annotation. Various modules enhance the reliability and response time of the web service using the concept of ontology. Thus from the perspective of architecture (Figure 1) the Ontology based Service Index Annotator system is divided into main parts domain ontology and creation of web service from the programmer and domain based. The verification of semantic web service module and the end result i.e. the generated output generated is tested using the annotated source code. The process is differentiated in two different sections, which include Domain based and web programmer. In the first part the domain section the focus is given to Ontology Based Service Index Annotator for the construction and generation of pool instance. In the next phase the input and output is introduced to execute to the test cases. In the second part, the web programmer the WSDL file is used to support the annotation. In the next stage the execution of test is conducted[28].

The main phase of semantic annotation creation includes the Domain ontology construction, test cases, WSDL file, annotation, discovery, test case execution, defection detection, curating and deploying. In the research the main direction of verification of semantic web service module is explored using various semantics, which are presented in eight different phases.

STAGES:

Stages 1: Creating online shopping and WSDL creation

Stages 2: Ontology construction for online shopping and extending WSDL

Stages 3: Test case generation with legal and illegal values for annotations

Stages 4: Execution of test cases

Stages 5:Verifying outcome of testing.

Stages 1: Creating online shopping and WSDL creation

Creating online shopping and WSDL creation

Web service is created for online shopping. The application includes operations such as login, product details, purchase products, product delivery and logout. The operations are described as service descriptions in web service through WSDL file. The WSDL stands for Web Service Description Language, which describes the operations input and output parameters. The operations input and output are based on the methods used in web service. The WSDL files are useful in service discovery and service publishing.

Stages 2: Ontology construction for online shopping and extending WSDL

4.2. Ontology construction for online shopping and extending wsdl

Ontology is constructed based on the online shopping domain concepts. The ontology consists of classes and properties. The output of WSDL file created from the previous module is given as input for semantic annotation. The WSDL file of the online shopping is semantically annotated using the classes of the ontology. The WSDL file is now semantically annotated with the ontology classes which in turn consist of operations input and output.

4.3. Annotation Verification Process

In the Annotation Verification Process in the initial phase the i.e. Phase 1 is where the instance of each annotation is prepared which turns into a pool of annotated instances, in Phase 2 the generation of these annotated instances pool the parameters for input test cases are created. Thirdly, these constructed parameters are tested and executed in Phase 3and in phase 4 the defects detection in the annotation is carried out from the results parts. Later in phase 5 the parameters for output test cases are exploited from the executed results and generated output test cases are executed in phase 6 and in phase 7 the analysis for identifying the defects is performed. A curator can then correct annotations in the light of the defects discovered (phase 8). The following section illustrates the same

Stages 3: Test case generation with legal and illegal values for annotations

The test cases are created using the automatic process, in which researcher specifies and test the registry of semantic annotation and parameter set. Moreover, the ontology location for semantic annotations is shown below. It is evident that the legal and illegal values are not illustrated in the algorithm for selected parameter (variable pool). Thus the annotated instances should be associated with both legal and illegal values. In addition, during the input parameter testing (op, p) identifies to be

associated with a precondition prec and minimize the legal value number during the annotation accuracy verification of input parameter which satisfies the predicate prec. An operation op has a set of input parameters given by the function inputs(op) and a set of outputs given by outputs(op).A parameter is denoted by a pair op, p where p is the name of the parameter and op is the operation to which the parameter belongs. Especially, while testing the annotation of op, p iff holds(prec, op, p,v), where holds(prec, op, p,v) is true iff prec is satisfied when v is used as, were v stands for legal value. The strong semantics are used to create the test cases. For execution, the annotation should have some pre-condition in strong semantics. Once the pre-condition is satisfied, the annotation is selected as input or output test case. The concept of pre-condition provides high level of accuracy for identifying the test case belongs to legal or illegal.The algorithm used for generating the test cases is as follows:

Input psi = input params to be tested

Output psi = empty test suite

Begin

For each operation parameter <op, p>∈ psi

pool := selected values from pool (should be a mixture of legal and illegal)

For each v in pool

If v is legal Select tuple of legal values lv for other input params of this operation such that the precondition of op holds

eo := “normal”

If v is illegal

Select tuple of legal values lv for other input params of this operation

eo := “abnormal”

tc := <<op, p>,lvU v, eo>

Add test case tc to tsi

End

Stages 4: Execution of test cases

The operation is tested with input and output test cases and executed. The execution of test cases terminated with either normal or abnormal termination. In the execution phase the SoapUI [29]is adoptedto support the large web service invocations.The algorithm which is used during the executing test Cases for Input Parameters is as follows.

Input: tsi = test suit

Outputs rs = a null tuple

InputLog = the results of execution of test cases

output Log = the results delivered byoutput parameter

Begin

For each tc in tsi

tc = <op, pi>, vs, eo>

Execute operation op with input params vs

rs := result returned from op

If op terminated normally then

ao := “normal”

For each output param op, p of op

r := value for <op, p> in rs

Add <<op, p>,r> to output Log

Else

ao := “abnormal”

Add <op, vs, rs, ao> to input Log

End

The normal termination describes the values is legal and abnormal termination of test case describes the value is illegal. However, in real time interpreting the web service’s normal and abnormal terminations is more complex and in the abnormal terminations case various other factors affect the web service to exit with an exception, other than providing illegal input.

Stages 5: Verifying outcome of testing.

In order to construct the output parameters from the test cases of input parameter execution i.e. the outcome of test case execution are used. Thus to generate the output parameter’s Test Cases the following algorithm is used.

Input pso = output params to be tested

Outputs rs = empty test suite

Begin

For each operation parameter <op, pi> € pso

os := output values for <op, pi> from output log

For each ov in os

Find all services ss with input annotated with the same concept as <op, p>

Select tuple of legal values lv for other input params of this operation

eo = = expected outcome for lv ∪ ov

tc = <<op, pi> lv ∪ ov, eo>

Add test cases tc to tso

End

The generated output parameter's Test Cases is tested and interpreted using the following algorithm is used.

Input tso = test suite for outputs

Outputs output Log = the results of execution of test cases

Begin

For eachtc in tso

tc := <<op, pi>, vs, eo>

Execute operation op with input params vs

rs := result returned from op

If op terminated normally then

ao := 'normal'

Else

ao := 'abnormal'

Add <op, vs,rs, ao> to output test Log

End

In addition, the executed results are logged. The logged results are compared with the expected outcome. If the results resemble the expected outcome, then the annotation is verified as valid. If the results contradict the expected outcome, then the annotations are considered as error. The defect is detected by performing above steps. The errors are rectified in semantic annotations.

5. Test Adequacy Criteria

In the semantic annotation process is a explicate Adequacy Criteria or underlying principle is a means which provides form or strategy to compare the different integrating strategies[30]. In the present study the logged results are compared with the expected outcome. If the results resemble the expected outcome, then the annotation is verified as valid. If the results contradict the expected outcome, then the annotations are considered as error. The defect is detected by performing above steps. The errors are rectified in semantic annotations.

6.0 UML Diagrams

6.1. Class Diagram

A class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code.

Activity Diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. Activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

6.2. Use Case Diagram:

A use case diagram is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

6.3. Sequence Diagram

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams typically are associated with use case realizations in the Logical View of the system under development. The Sequence Diagram models the collaboration of objects based on a time sequence. It shows how the objects interact with others in a particular scenario of a use case.

7. RESULTS

In this paper, comparison was made on the present proposed service parameters testing using semantically annotated WSDL and using non- annotated WSDL, i.e., by using the ontology constructed proposed for the domain of services and without using ontology. The basic objective behind adding semantics to WSDL is to enhance the use of metadata that comes in the form of ontology. In this work, the ontology adds constraints to all the service parameters thereby during the testing phase the metadata related to the service parameters can be utilized for data validation of the parameters, thereby reduces the code required to be written for data validation and also reduces the number of test cases used in the testing phase of the project.

In the following Table1, for the operation registration which has name, age, address, mobile number and email id as the input parameters, nearly 5^3=125 test runs are required to test the operation completely with the assumption that each parameter may have 3 constraints. Also the developer has to take care of all these constraints in the data validation code and appropriately take actions for the constraints. This increases, not only the development time of the software and also testing time of the service both from the provider and from the consumer. But if the constraints specified are taken from the ontology for testing, then it can be automatically tested during testing phase and throws the corrective measures to the provider and also to the consumer thereby reducing the test runs to a great optimum level.

8. CONCLUSION

The proposed system provides accurate service results of semantic web service annotation before deploying in public use. Verification process includes conventional testing methodology for testing semantic annotations. Annotated instances are taken as test cases and test cases are executed with legal and illegal values. The resulted logged values are matched with expected outcomes. The outcomes may be normal and abnormal termination. If abnormal termination takes place, the annotation is corrected. The system provides high defect detecting power.

References

[1] D. Schrimpsher and L. Etzkorn, “A model to predict quality of a reduced ontology for Web service discovery on mobile devices,” Knowl. Eng. Rev., vol. 29, no. 02, pp. 201–216, Mar. 2014.

[2] P. Lord, P. Alper, C. Wroe, and C. Goble, “Feta: A Light-Weight Architecture for User Oriented Semantic Service Discovery,” Semant. Web Res. Appl. Lect. Notes Comput. Sci., vol. 3532, pp. 17–31, 2005.

[3] A. Heb and N. Kushmerick, “Automatically Attaching Semantic Metadata to Web Services,” in Proc. IJCAI Workshop Information Integration on the Web, 2003.

[4] D. Jannach, K. Shchekotykhin, and G. Friedrich, “Automated ontology instantiation from tabular web sources-The All Right system, Web Semantics: Science,” Serv. Agents World Wide Web, vol. 7, no. 3, pp. 136–153,, 2009.

[5] A. Q. M. Salih, “Towards From Manual to Automatic Semantic Annotation: Based on Ontology Elements and Relationships,” Int. J. Web Semant. Technol., vol. 4, no. 3, pp. 21–36, 2013.

[6] H. N. Talantikite, D. Aissani, and N. Boudjlida, “Semantic annotations for web services discovery and composition,” Comput. Stand. Interfaces, vol. 31, no. 6, pp. 1108–1117, Nov. 2009.

[7] M. Paolucci, T. Kawamura, T. R. Payne, and K. Sycara, “Semantic Matching of Web service Capabilities,” in Proceedings of the First International Semantic Web Conference, 2002, p. Proceedings of the First International Semantic We.

[8] P. A. Bonatti, C. Duma, N. Fuchs, W. Nejdl, D. Olmedilla, J. Peer, and N. Shahmehri, “Semantic web policies - a discussion of requirements and research issues,” in 3rd European Semantic Web Conference (ESWC),volume 4011 of Lecture Notes in Computer Science, 2006.

[9] G. Denker, L. Kagal, T. Finin, M. Paolucci, and K. Sycara, “Security for DAML web services: Annotation and matchmaking,” in Proceedings of the 2nd International Semantic Web Conference, 2003.

[10] A. Ankolekar, F. Huch, and K. P. Sycara, “Concurrent Execution Semantics of DAML-S with Subtypes,” in ISWC ’02 Proceedings of the First International Semantic Web Conference on The Semantic Web, 2002, pp. 318–332.

[11] B. Medjahed, A. Bouguettaya, and A. K. Elmagarmid, “Composing Web services on the Semantic Web,” VLDB J. Int. J. Very Large Data Bases, vol. 12, no. 4, pp. 333–351, Nov. 2003.

[12] A. A. Patil, S. A. Oundhakar, A. P. Sheth, and K. Verma, “Meteor-s web service annotation framework,” in Proceedings of the 13th conference on World Wide Web - WWW ’04, 2004, p. 553.

[13] K. Gomadam, A. Ranabahu, M. Nagarajan, A. P. Sheth, and K. Verma, “A Faceted Classification Based Approach to Search and Rank Web APIs,” in 2008 IEEE International Conference on Web Services, 2008, pp. 177–184.

[14] E. M. Maximilien and M. P. Singh, “A framework and ontology for dynamic Web services selection,” IEEE Internet Comput., vol. 8, no. 5, pp. 84–93, Sep. 2004.

[15] S. B. Davidson and J. Freire, “Provenance and scientific workflows,” in Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD ’08, 2008, p. 1345.

[16] B. Jiang and Z. Luo, “A New Algorithm for Semantic Web Service Matching,” J. Softw., vol. 8, no. 2, pp. 351–356, Feb. 2013.

[17] A. BOUKHADRA, K. BENATCHBA, and A. BALLA, “Automatic composition of semantic Web services-based alignment of OWL-S.” [Online]. Available: http://ceur-ws.org/Vol-867/Paper39.pdf. [Accessed: 09-Apr-2015].

[18] V. Saquicela, L. M. Vilches-Blazquez, and O. Corcho, “Lightweight Semantic Annotation of Geospatial RESTful Services.” [Online]. Available: http://oa.upm.es/6968/1/Lightweight_Semantic.pdf. [Accessed: 09-Apr-2015].

[19] A. Ranabahu, P. Parikh, M. Panahiazar, A. Sheth, and F. Logan-Klumpler, “Kino: A Generic Document Management System for Biologists Using SA-REST and Faceted Search,” in 2011 IEEE Fifth International Conference on Semantic Computing, 2011, pp. 205–208.

[20] A. A. Patil, S. A. Oundhakar, A. P. Sheth, and K. Verma, “Meteors web service annotation framework,” in WWW, 2004, pp. 553–562.

[21] A. Ankolekar, M. Burstein, J. R. Hobbs, O. Lassila, and D. Martin, “DAML-S: Web service description for the semantic web,” Proceedings of the 1st International Semantic Web Conference Sardinia, 2002. .

[22] G. Li, S. Deng, H. Xia, and C. Lin, “Automatic Service Composition Based on Process Ontology,” IEEEXplore,, 2007.

[23] Y. Song, L. Liu, and P. Ren, “Web Service Composition Based on the Annotated Ontology,” in Fifth International workshop on education technology and computer science, 2009.

[24] A. Maedche, B. Motik, N. Silva, and R. Volz, “MAFRA - A MApping FRAmework for Distributed Ontologies,” in EKAW ’02 Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 2002, pp. 235–250.

[25] A. Naumenko, S. Nikitin, V. Terziyan, and A. Zharko, “Strategic industrial alliances in paper industry: XML-vs Ontology-based integration platforms,” Learn. Organ., vol. 12, no. 5, pp. 492–514, 2005.

[26] S. Mokarizadeh, P. Küngas, and M. Matskin, “Ontology learning for cost-effective large-scale semantic annotation of XML schemas and Web service interfaces,” EKAW, vol. 10, pp. 401–410, 2010.

[27] B. Meyer, “Seven Principles of Software Testing,” IEEE Comput., vol. 41, no. 8, pp. 99–101, 2008.

[28] M. Sabou and J. Pan, “Towards Semantically Enhanced Web Service Repositories,” Web Semant., vol. 5, no. 2, pp. 142–150, 2007.

[29] EVIWARE, “SoapUI; the Web Services Testing tool.” 2015.

[30] C. Eschenbach and M. Gruninger, Formal Ontology in Information Systems: Proceedings of the Fifth International Conference (FOIS 2008). IOS Press, 2008, p. 325.