Journal of Engineering and Applied Sciences 13 (9): 2791-2798, 2018
ISSN: 1816-949X

Ontology Based Annotation Verification Framework for Semantics Enabled Web Services

[1]S. Shridevi and [2]G. Raju
[1]Vellore Institute of Technology, Vellore, India
[2]Kannur University, Kannur, India

Abstract: Present scenario of technology is awaiting an automated annotation in order to discover an appropriate web service. Revealing a web service in a systemized form is difficult and hence the retrieval of an accurate web service is almost unfeasible. Prevailing web service annotation mechanisms suffers from a limitation such as compromise in quality of annotating a web service in a semantic manner using third party annotators and hence, annotating a web service to maximum appropriateness is almost unfeasible and a non-trivial task too within stipulated time. In order to resolve this and to acquire an appropriate web service with an utmost relevancy, a semi-automatic framework is developed in this research by means of employing Semantic Annotations for WSDL and XML Schema (SAWSDL) annotation mechanism utilizing ontology based annotations and the annotations are verified for its correctness and defects are detected and curated.

Key words: Ontology, semantic web services, annotation, Web Service Description Language (SAWSDL), annotations, party

INTRODUCTION

The concept of the semantic web can be extended to web services to envision semantic web services. Web services provide only syntactic interoperability because although the syntax and structure of the exchanged data is verified, the information content is not "understood". Semantic annotations of web services are useful in service discovery only when it reflects accurate services. Manual annotation requires experts in domain and is time consuming. As annotation is performed by the third party, misinterpretation may take place. This is due to the annotator neither involved in the domain ontology construction and development of web service. False positives (i.e., errors are not revealed as defects) are high. No techniques are used for verifying semantic web service annotations. The main objective of the research is adding semantic annotations to facilitate service discovery, to provide a simple framework for the implementation of semantic web services life cycle and their diffusion in industrial settings, to focus on functional aspects of services by adding semantics to it and providing low cost tool for verifying semantic annotations (Shridevi and Raju, 2016) and mainly using Ontology-based test adequacy criteria to minimize false positives and to evaluate the performance evaluation for complete life stages of semantic web service in a peer to peer network.

Literature review: Semantic annotation plays the major role in the web service annotation for extracting the relevant services from the user request via. the internet. In the recent years, web service discovery has been interfaced with ontology based semantic annotation that indicated the accurate service results (Puttonen *et al*., 2013). Certain shortcomings of the web service implementations are given as follows.

The Quality of Service (QoS) was affected due to the combination of the several properties or qualities of services such as availability (service operating time), response time (processing time for requests) and throughput. The occurrence of semantic problems generated in the web service annotation like content availability, ontology performance, measurability, multilingualism, loaded large information, constancy services affects the discovery process of services. The occurrence of composite problems like irrelevant information, the processing time for execution and accuracy.

The literature review aims to explore the potential challenges associated in evaluating the performance of semantically annotated web services throughout its life cycle. The motive is to give the reader with the state of the art offered technologies and ontology languages able to support the method of annotation of WS with semantically-enriched data (Belyaeva *et al*., 2015) for each functional and non-functional aspect and measure the

Corresponding Author: S. Shridevi, Vellore Institute of Technology, Vellore, India

performance of such annotated web services throughout its life cycle. Functional annotation is bothered with tagging and enriching the specification of the functionalities exposed by the WS with semantic data whereas non-functional annotation is concerned with the specification of the service with SLA and QoS related information. In our survey, we mainly focussed in implementation and performance of annotation evaluation aspects.

Zhang *et al.* suggested an automated methodology for semantic annotation of web services, focused on the DBpedia knowledge which is available as linked data. A novel semantic annotation methodology was proposed for the inputs and outputs of web services by making use of DBpedia dataset and DBpedia Spotlight and evaluation of approaches were performed. The experimental results exhibited the approaches to be effective and feasible for service annotation. The ultimate aim of the research was to enhance the web services along with the semantics of linked data.

Bouchiha and Malki (2012) illustrated all benefits of web services technology by annotating WSDL syntactic descriptions of web services by ontological models. The recommended technique basically relied on an annotation methodology which comprised of the two distinct phases like categorization phase and matching phase. Categorization phase-categorised the WSDL elements into equivalent domain. Matching phase-allowed association of entity from WSDL elements.

The annotation process focused on ontology matching techniques which used certain similarity measures to evaluate its performance.

Tahir *et al.* (2013) reviewed the systematic procedure of the semi-automatic semantic annotation of the web services. This research mainly considered the semantically annotating web service aspects and available state-of-the-art analysis of the formal and precise approach. Then three main scenarios were focused in this research work were given as follows:

- Precise and adopted ontology
- Semantic services to implement the complex and industrial sides
- Annotation of semantic services was supported by the tools and selected with match-making facilities

Wang *et al.* surveyed the semantic web service formalisms along with the existing approaches like WSMO, OWL-S and SAWSDL were compared and the required results were described in this survey research. The OWL-S determined the agent planning approach and demonstrated the web services as processes. The three distinct frameworks and associated context-dependent were addressed in this research for different viewpoints on semantic web services. The chief highlight of the OWL-S contained the explicit definitions of choreography and orchestration enabling effective resources of computational usage.

Furno and Zimeo evaluated the semantic web services with the help of context aware compositions. The proposed approach utilized to automatically generate the context-aware composition from the design of context aware services that can be exploited as a flexible domain. The context representation (defined by designers or implicitly inferred by the system) included in the problem definition of an extended tool extended the services by improved the precision of automatic compositions and model reference with an example as well as depicted the clear advantages of the proposed model. The definition of the model was emphasized by conducting numerous analyses and addressing the problem happened in the system. The composite solutions were validated by planning and evaluating the rules along with the problem expansion of exploited context. They indirectly excluded during domain construction if, for an abstract service, the context fitted by using one of the post conditions but only one of them satisfied the pre-conditions of the next services (e.g., there is disk space for retrieving a HD file but the player was not able to play). While applying the context rules, the state dependencies were accounted during planning action and contextualized expansion of services. Further, they enhanced the approach by extending the model and the tool.

Jiang and Luo (2013) proposed a new algorithm for semantic web service matching and considered the major problem of web service matching such as interoperability, absence of semantic evidence. They recommended the semantic annotation framework for describing the web service in supporting the semantic web service by an extended UDDI service. Hence, the service discovery efficiency was improved by means of a novel algorithm based on the semantic similarity computations.

Tahir *et al.* (2013) reviewed the functional testing of semantic web services and summarized results of the current state of the art of research by providing the responses to a researched queries. The 5 well-known digital libraries of the automatic searching in the pre-defined procedures were analyzed. Then, the selection based on the relevant category of the obtained results was obtained from the 34 identified studies. Finally, the required information were organized and summarized as an added advantage. Therefore, the methodology was checked whether the approaches reached out its credibility. The proposed approach improved the current state of research in testing the semantic web services.

Jokhio *et al.* (2017) studied several techniques for deriving the test cases from semantic web service descriptions with the help of Web Service Modelling (WSMO) framework. Test cases were produced by model checking with computed trap properties from coverage procedures. Here, the model based test case generation and code based evaluation techniques which remained independent of each other to provide accurate measures of testing results. The excellence of test cases was measured in terms of capacities to determine the injected faults at the code level. They also implemented the tool to automate the steps for the mutation based evaluation. It was resolved that the procedure was proficient of spontaneously producing an effective set of test cases from the WSMO goal descriptions.

Rodriguez-Mier *et al.* evaluated the integrated semantic web service discovery and offered the graph-based outline for a semantic input-output parameter correspondent of services interfaces. The service discovery with dependency was verified by the theoretical method of service composition. Integrated service discovery and matchmaking with composition process was efficiently performed based on the framework. Based on two pre-existing separate components like iServe and ComposIT developed reference implementations in this framework. The service composition impacts of discovery and matchmaking were empirically studied with the help of reference implementations. The varying performances were obtained based on the three different configurations provided by the service composition. This led to the execution time on the service composition to reduce and discussed the empirical analysis depicted indeed with discovery engines followed on typical approaches.

The prevailing issues in the existing algorithms were investigated and discussed in this section. Existing web service annotation contrivances suffers from following limitations:

- Lack of quality in annotating a web service in a semantic manner via manual method
- Annotating a web service to an extreme appropriateness is practically unfeasible
- Failed to test the actual behavior of the web service instead of testing annotated functionality
- It is problematic to attain an operative set of inputs as well as output parameters

Hence, given a collection of semantic annotations for web services, our goal is to verify and improve the annotations to support web service interoperations. We significantly included the ontology validation into account, so that, the semi-automatic bootstrapping of ontologies would recommend potential correct concepts from knowledge base.

MATERIALS AND METHODS

Proposed framework and its implementation: In order to resolve these issues and to acquire a web service with an utmost relevancy a semi-automatic framework is developed in this research by means of employing SAWSDL annotation mechanism to utilize ontology based annotations and correlate those with the equivalent web services. The proposed web service annotating architecture is capable of validating and verifying those web services that are annotated using ontology mechanism (Gonen *et al.*, 2015). The fundamental intention involved in designing this architecture is to accomplish a precise annotation for the web service in a semantic manner. Figure 1 illustrates the overall flow of the proposed Semi-Automatic Semantic Annotation and Verification (SASAV) architecture.

Ontology development and validation model: A diagnostic task is employed to validate the generated ontology (Gangemi *et al.*, 2006; Raad and Cruz, 2015) by means of considering the processes, task oriented elements and the attributes associated with the structure concerned as illustrated in Fig. 2. Ontological descriptions defined on the basis of quality and:

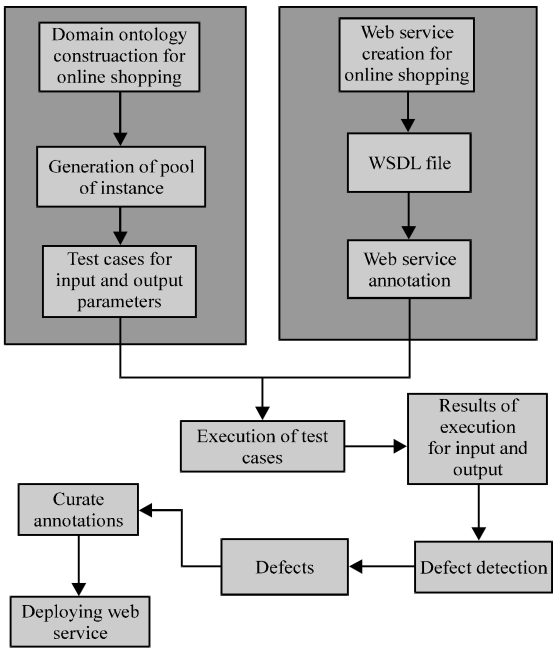

Fig. 1: SASAV architecture

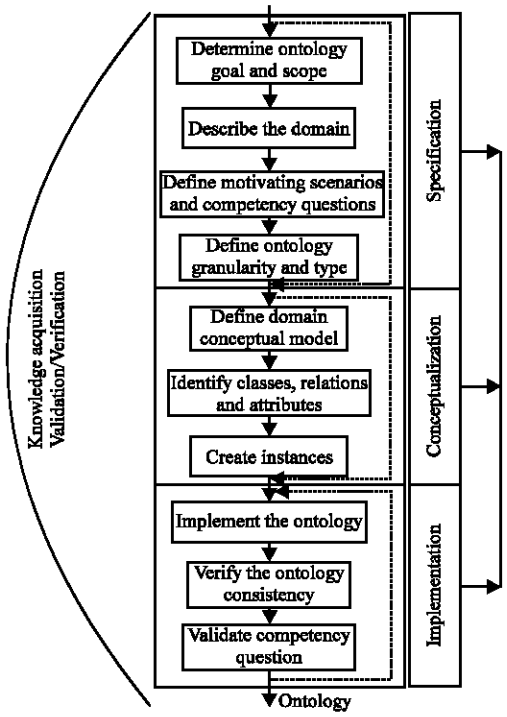

Fig. 2: Ontology development and validation

- Value spaces
- Evaluating fitness of ontology through pre-defined principles
- Parameters defined across principles
- Parameter dependencies
- Preferential ordering
- Trade-offs

All these parameters are considered while constructing the ontology. Though these validating approaches (Li *et al*., 2009) tend to crosscheck the competency of the ontology constructed, it is not sufficient to analyse the complete activity of the web service framework concerned as the primary motive of the proposed research is to verify and test the annotations success with respect to web services. Hence, it is validated through test case implementation approach discussed in the next study.

Semantic Annotations for WSDL and XML Schema (SAWSDL): The key purpose of describing a web service through utilization of WSDL is to actually represent an appropriate meaning of a web service that is designated in its syntactic level instead of verifying the outward look of the messages or operations involved in a web service. The constituents of both WSDL and XML schema are indicated by means of deploying an extended WSDL layer after retrieving its associated semantics (Farrell and Lausen, 2007). Thus, operations involved in a web service, input as well as output messages and annotations of those appropriate WSDL interface are defined within the XML schema. The extensions provided by SAWSDL are distributed into two different methods given as schema mappings designated for transforming information between XML structure and its connected semantic model and secondly the model reference indicating those semantic concepts.

Fundamentally, the process of annotation utilizes a stemming methodology into the prevailing WSDL document and that is prolonged for creating an appropriate ontology in search of a reference Algorithm 1.

Algorithm 1; Algorithm for abstracting meaningful word from WSDL element:
Input: Name of the WSDL element
Output: meaningful word
Step 1: Perform tokenization on all WSDL elements by recognizing the capital letter
Step 2: If the observed token is in the list of unused words then remove it from the list of useful tokens
Step 3: If the observed token is existing within the list of meaning full words then merge this token also with it
Step 4: return meaningful word

Process of annotating a web service is done by means of getting concepts from those ontologies constructed in appropriate manner Algorithm 2.

Algorithm 2; Algorithm for getting annotation for the element:
Input: WSDL Element
Output: OntologyURI annotation for all i/o parameters.
Step 1: Initiate ontologyURI to be null
Step 2: Obtain meaningful word from the WSDL elements
Step 3: Obtain matchType by assessing the ontology component of every element
Step 4: On recognizing a matchType attribute to be true then generate a model reference and merge that particular WSDL element
Step 5: If the matchType attribute is computed to be false then add that WSDL element to list of unused words
Step 6: Return OntologyURI
Step 7: Repeat steps 1-6 for all meaningful words of WSDL

After recognizing the matching criterion from the attribute responsible for getting the ontology based annotation, the model reference for the parameter is generated. Finally, a SAWSDL file is generated from those abstracted WSDL elements by using the following Algorithm 3.

Algorithm 3; Algorithm for generating SAWSDL from WSDL:
Input: WSDL file
Output: SAWSDL file

Step 1: Obtain WSDL file from the server
Step 2: Validate WSDL
Step 3: Obtain operation names from the WSDL file
Step 4: Process each operation name for its input and output parameters
Step 5: Signify operations with modelreference attribute to annotate the input and output parameters with the respective Ontology URIs.
Step 6: Repeat step 5 to annotate all required distinct input and output parameters of the service operations and
Step 7: Exit when SAWSDL annotations are generated.

SAWSDL is concerned with mapping elements from WSDL to a higher conceptual level, i.e., the ontology level. To achieve this SAWSDL defines the following extension attributes (Farrell and Lausen, 2007).

Model reference-associates, via. a URI, a WSDL component with a semantic concept. This attribute is typically used on element declarations, type definitions, interfaces (or port types) and operations. Lifting schema mapping-added to element declarations and type definitions for specifying the mapping from XML to the semantic layer. Lowering schema mapping-added to element declarations and type definitions for specifying the mapping from the semantic layer to XML.

The lifting schema mapping and lowering schema mapping attributes support data mediation between web services. Take for example, two Web services that are trying to communicate where the output from the first does not match the input of the second. The output from the first Web service can be lifted to the ontology layer by its lifting schema mapping. The second Web service which is pointing at the same ontology concept, provides a lowering schema mapping that lowers from the ontology to the input format it requires.

Test case generation for the input parameters of the web services: Here, the test cases are created by the client for all service operations and then it is validated by evaluating the input parameters with appropriate test cases (Javed *et al*., 2016; Rusli *et al*., 2011). Then, the input parameters are verified in accordance with the web services and the legal value and illegal values that are associated with all service operations were logged as per below algorithm. Algorithm 4 used for generating the test cases is as follows:

Algorithm 4; Generation of test cases:

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>, lv∪v, eo>
Add test case tc to tsi
End

Test case generation of the web services for the output parameters: Based on the normal and abnormal termination of the test cases, the authorized decision is taken for executing the test cases. The test cases are executed and preserved with the results in an input log and its respective outputs obtained for each and every execution are stored in the output log. Algorithm 5 which is used during the executing test cases for input parameters is as follows.

Algorithem 5; Input parameters:

Input: tsi = test suit
Outputs rs = a null tuple
Input Log = the results of execution of test cases
output Log = the results delivered by output 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

Test case generation of the annotated services for the input and output parameters: Annotation is defined as the appropriate description of the web services that are provided by the service provider. The accuracy of the input and output parameters are validated by generating the specific test cases for both annotated input and output parameters. The input instances that are not linked with any of the semantically annotated web services are removed from the overall test cases. Thus, to generate the output parameter's test cases the following Algorithm 6 is used:

Algorithm 6; Annotated web service:

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 are tested and interpreted using the following algorithm.

```
Input tso = test suite for outputs
Outputs output Log = the results of execution of test cases
Begin
For each tc 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.

Service discovery in peer-to-peer network: The architecture allows each and every peer to act like a service requester and provider where all types of services that are involved are semantically annotated using ontology and evaluated and described within every distinct peer that exists within a servicing architecture. Also, it has the responsibility for exploring the equivalent web services as per the request projected (Fig. 3).

The accuracy of web services deployed on the server side is validated by means of generating test cases on the client side. Providing semantic annotations by utilizing ontological structures typically describes the functionality of a web service. The correctness of those annotations along with the specific behaviour of web services is verified through utilization of conventional testing strategy. Web services are specified by distinct operations along with its associated parameters involved. Input parameters and output parameters involved in executing a functionality of a web service are included within the test cases concerned and are validated. Those test cases defined valid are opted to form a test suite and those test suites are executed to validate the consistency of the entire web services being created. The test cases for annotated instances that are executed until the terminating point tend to consists of legal values and functionality of that specific test case is defined normal. If the test case concerned is terminated in an unusual manner, then it is declared as abnormal. The storage logs are utilized to store these results in a separate manner given as inputLog and outputLog, respectively. On realizing the defect in annotations, a curator is employed to correct those faults and the proposed methodology realizes accurate service results for those semantically annotated web services.

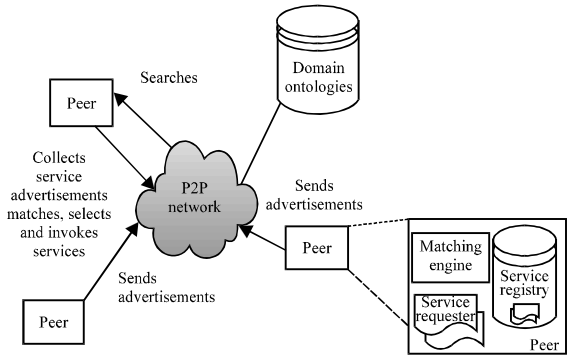

Fig. 3: Peer to peer network

In our research, an ontological framework is developed and consistencies and validation of those annotations of respective ontologies are verified and defects are detected and curated manually.

Implementation: SAWSDL based SOAP web service application was created for online shopping using Java and Jena API. The application includes operations such as login, product details, purchase products, product delivery and logout. All services required were created and few are listed in Fig. 4 and 5.

- @WebMethod() public void Payment (int cardnum, String mailid, String bankname, long amount)
- @WebMethod() public String ProductDelivery (long s1, long s2)
- @WebMethod public String registration (String uname, String uid, String pass, String address, String city, String mobile, String mail)
- @WebMethod public String viewHeadPhone (Hid, Hname, Hbrand, Hprice, Hstock)
- @WebMethod public String viewKeyBoard (kid, kname, kbrand, kprice, kstock)
- @WebMethod public String viewlaptop (Hid, Hname, Hbrand, Hprice, Hstock)
- @WebMethod public String viewmodem (Hid, Hname, Hbrand, Hprice, Hstock)
- @WebMethod public String viewdigitalpen (kid, kname, kbrand, kprice, kstock)
- @WebMethod public String viewproducts (items)

Ontology is constructed based on the online shopping domain concepts. The semantic online shopping web service application was created in SOAP based web service architecture style for testing its annotations correctness during its execution.

RESULTS AND DISCUSSION

This study discusses the testing procedures and its results obtained for each and every test cases being

Fig. 4: Ontology construction

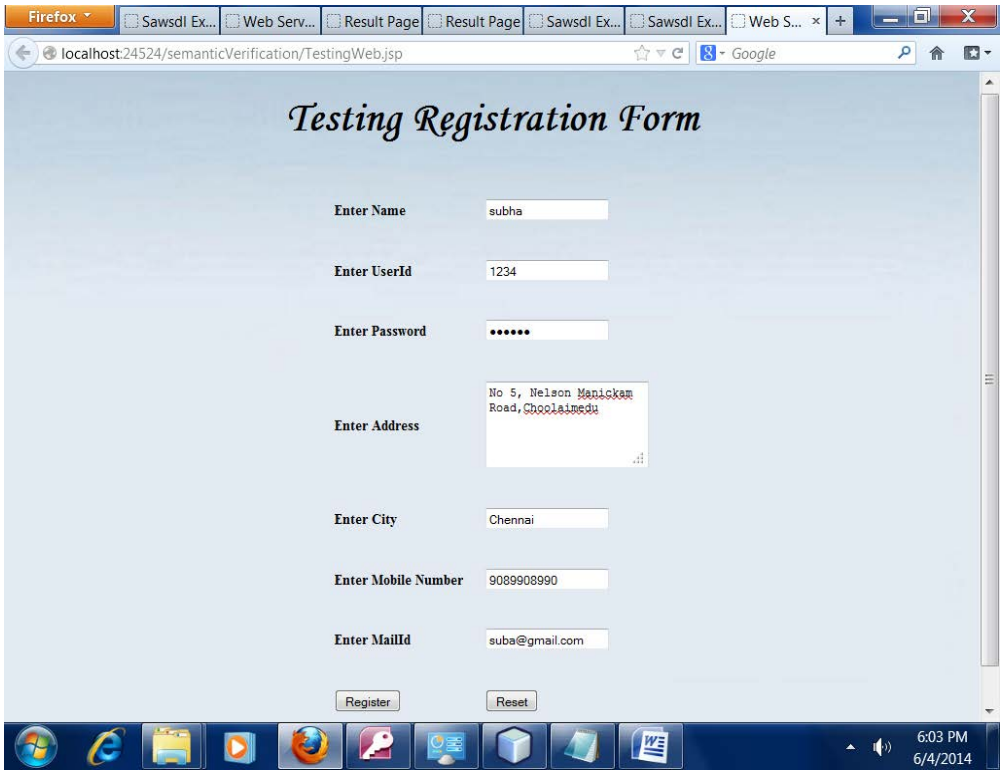

Fig. 5: Testing registration web service

implemented for both inputs as well as output instances. The appropriateness of the web services is revealed through the deployment of test cases for every input and output instances specified. Input requests are completely matched with appropriate output instances in order to process the user request with extreme correctness. If there is any mismatch occurs in between those attributes inferred and its instances, then, the functionality of that web service concerned is wrong and it is meant to be rectified for the defects. This sometimes causes the test case to get blocked due to the dependency found between them. The defects are revealed and the test cases

Table 1: F-measure of proposed framework

Services	Parameters	Test cases	Fault cases	Noor passed	Noor faied	F-score
1	3	9	3	5	4	0.86
2	4	12	5	6	6	0.91
3	5	15	4	8	7	0.73
4	3	9	5	2	7	0.83
5	4	12	7	4	8	0.93
6	4	12	7	4	8	0.93
7	2	6	3	2	4	0.86
8	3	9	3	5	4	0.86
9	4	12	6	4	8	0.86
10	5	15	5	8	7	0.83

of those web services that got executed completely are termed as error free and functionality of those web services are tend to be perfect. Here, different measures such as precision, recall, F-measure, test case selection time, error identification time, test cases executed, test cases passed, test cases failed, test cases blocked, defect density, Defect Removal Efficiency (DRE) and defect leakage are used to validate the performance. The precision is defined as the relevancy between the web services that is assessed against the ratio of retrieved services. Recall is defined as a measure of retrieving the services against the ratio of retrieved relevant web services. The F-measure is computed by integrating both precision and recall measures. Table 1 illustrates the precision, recall and f-measure of proposed annotation detection methods. From the evaluation, it is observed that the proposed Semi-Automatic Semantic Annotation and Validation (SASAV) framework provides the below F-measure values for defect detection when compared to the other techniques.

The test case execution is defined as the process of associating the test case that deployed to an expected outcome. The overall quality of the web service that is obtained based on the user requirement is known as the test case pass. The test case failed cannot proceed to the next level of execution. The execution of the test case is blocked when the sub-ordinate test case fails.

CONCLUSION

This research aims to develop a new framework for annotating the web services by using WSDL. In future, this research will be enhanced by validating the procedure of web services in a completely automated fashion with the use of centralized registry for service discovery. Also, the efficiency of service execution can be enhanced based on the resource optimization characterization.

REFERENCES

Belyaeva, E., A. Kosmerlj, A. Muhic, J. Rupnik and F. Fuart, 2015. Using semantic data to improve cross-lingual linking of article clusters. Semant. Sci. Serv. Agents World Wide Web, 35: 64-70.

Bouchiha, D. and M. Malki, 2012. Semantic annotation of web services. Proceedings of the 4th International Conference on Web and Information Technologies (ICWIT'12), April 29-30, 2012, College of Literature, Sidi Bel-Abbes, Algeria, pp: 60-69.

Farrell, J. and H. Lausen, 2007. Semantic annotations for WSDL and XML schema. World Wide Web Consortium, Cambridge, Massachusetts, USA.

Gangemi, A., C. Catenacci, M. Ciaramita and J. Lehmann, 2006. Modelling ontology evaluation and validation. Proceedings of the Conference on European Semantic Web (ESWC'06), June 11-14, 2006, Springer, Budva, Montenegro, ISBN:978-3-540-34544-2, pp: 140-154.

Gonen, B., X. Fang, E. El-Sheikh, S. Bagui and N. Wilde *et al*., 2015. Ontological support for the evolution of future services oriented architectures. Trans. Mach. Learn. Artif. Intell., 2: 77-90.

Javed, H., N.M. Minhas, A. Abbas and F.M. Riaz, 2016. Model based testing for web applications: A literature survey presented. J. Software., 11: 347-361.

Jiang, B. and Z. Luo, 2013. A new algorithm for semantic web service matching. J. Software., 8: 351-356.

Li, Z., M.C. Yang and K. Ramani, 2009. A methodology for engineering ontology acquisition and validation. AI EDAM, 23: 37-51.

Puttonen, J., A. Lobov and J.L.M. Lastra, 2013. Semantics-based composition of factory automation processes encapsulated by web services. IEEE. Trans. Ind. Inf., 9: 2349-2359.

Raad, J. and C. Cruz, 2015. A survey on ontology evaluation methods. Proceedings of the 7th International Conference on Knowledge Engineering and Ontology Development (KEOD'15), November12-14, 2015, INSTICC, Lisbon, Portugal, pp: 179-186.

Rusli, H.M., S. Ibrahim and M. Puteh, 2011. Testing web services composition: A mapping study. Commun. IBIMA., 2011: 1-12.

Shridevi, S. and G. Raju, 2016. A novel approach for web service annotation verification and service parameters validation using ontology. Proceedings of the 3rd International Symposium on Big Data and Cloud Computing Challenges (ISBCC'16), February 23, 2016, Springer, Cham, Switzerland, ISBN:978-3-319-30347-5, pp: 399-415.

Tahir, A., D. Tosi and S. Morasca, 2013. A systematic review on the functional testing of semantic web services. J. Syst. Software, 86: 2877-2889.