# ENHANCED QUALITY OF SERVICE (QOS) FOR SIMILARITY COMPUTING IN WEB SERVICE DISCOVERY

*Chitra Devi. P [1] Anantharaj.B [2]*

*[1]PG Scholar, Department of Computer Science, Thiruvalluvar College of Engineering and Technology, chitradevi9711@gmail.com*
*[2]Head of the Department, Department of Computer Science, Thiruvalluvar College of Engineering and Technology, tcetcsehod@gmail.com*

**ABSTRACT**-This study aims to figure out the best possible web service is the one that completely fulfills the required functions while satisfying the QoS requested by a user. In this paper, we introduce a newsolution based on enhanced QoS (Quality of Service) for computing similarity exploiting both functional and non-functional user's requirements and providing the user ability to control and proceed the discovery of web services, i.e. the main aim of this work is to locate the appropriate web service correspondence with the context of the user by computing the similarity.

**Index Terms**--Web Service, Web Service Discovery, UDDI, WSDL, Quality of Service(QoS), Similarity Matching.

## 1. INTRODUCTION

A web service is defined a service offered by an electronic device to another electronic device, communicating with each other via the World Wide Web. In a Web service, Web technology such as HTTP, originally designed for human-to-machine communication, is utilized formachine-to-machine communication more specifically for transferring machine readable file formats such as XML (eXtensible Markup Language) and JSON (JavaScript Object Notation).

The Working Principle of Web Services are (i)The Service Provider publishes a description of the services it offers via the Service Registry. (ii)The Service Consumer could be a person or a program searches the service registry (Theregistry provides a central placewhere developers (Service Providers) canpublish new services or find existing ones.) to find a service that meets their needs. (iii)If the service is available in the registry, the Service Consumer uses the service offered by the Service Provider with the help of Binding process. If it is not available the registry intimates the non-availability of the requested service to the Consumer.

The contributions of Enhanced QoS for Similarity Computing inWeb Service Discovery are the followings:1) Proposing and developing a web service discovery framework that exploits both functional and non-functionalfeatures of web services. 2)The framework utilizes the SeqDisc approach to filter services determining the relevant services that represent functional requirement of the user. 3)Then, both contextual information of the user and relevant services are matched by measuring the similarity between them.4)These services are then ranked, which can in this context, achieve both functional and non-functional requirements of the user. 5)Conducting a set of experiments to validate the effectiveness of the proposed framework comparing it with three different methods.

Web Services are applied in the field ofIoT (Internet of Things),E-Commerce websites, Virtual Reality, Multimedia ,Web Development Companies

## 2. LITERATURE REVIEW

In the reference paper [1], a new and flexible approach to assess the similarity between WSs, which can be used to support a more automated WS discovery framework. The approach makes use of the whole WSDL document specification and distinguishes between the concrete and abstract parts. The concrete parts from different Web services have the same hierarchal structure, hence we devised a level-based matching approach. The abstract parts have different structures, therefore, we developed a sequence-based schema matching approach to compute the similarity between them.

The reference paper [2] introduces a mechanism that extends our Web Services Repository Builder (WSRB) architecture by offering a quality-driven discovery of Web services and uses a combination of Web service attributes as constraints when searching for relevant Web services. Our solution has been tested and results show high success rates of having the correct or most relevant Web service of interest within top results. Results also demonstrate the effectiveness of using QoS attributes as constraints when performing search requests and as elements when outputting results. Incorporating QoS properties when finding Web services of interest provides adequate information to service requestors about service guarantees and gives them some confidence as to the quality of Web services they are about to invoke.

The contribution of the reference paper [3] is two-fold. Firstly, a novel Web service discovery method based on the semantic similarity derived from the trained support-based kernel is introduced. We propose the creation of latent semantic kernel with the support-based algorithm using the concept of binning & merging, and then utilize the kernel to find semantically similar Web services for a user query. Secondly, a thoroughpractical experimentationand evaluation have been performed. The empirical analysis confirms that the proposed method is able to find semantic relations and thereby to improve the process of Web service discovery in comparison to traditional methods.

As in reference [9], they provided a brief survey of approaches for centralized and decentralized search of semantic web services. There are quite a few and sophisticated techniques for this purpose with various applications in different domains which make use of them in order to achieve a more precise service selection.

In reference [12], they provide an extensive review of semantic Web service discovery, highlighting the state-of-the-art approaches, the key semantic formalisms employed, as well as benchmarks and testbeds for performance evaluation. Defining a generic framework for semantic service discovery, we describe the key tasks and criteria involved in agent-based computing. A detailed comparison of the popular discovery systems is performed with a discussion on trade-offs between existing approaches.

The reference paper [13] proposes a new Web services discovery model in which the functional and non-functional requirements (i.e. quality of services) are taken into account for the service discovery. The proposed model should give Web services consumers some confidence about the quality of service of the discovered Web services.

The reference paper [15]proposes a generic model to represent QoS information in service advertisements as well as QoS requirements in service discovery requests. Based on this model, a matching algorithm and a ranking algorithm are presented. Given a service discovery request, the matching algorithm compares its QoS requirements with the QoS advertisements in the repository and

locates those services with matching QoS. The ranking algorithm furthers ranks these discovered services to facilitate the consumers to select services.

## 3. RELATED WORKS

A web service discovery is established with expressions like keyword, request, mapping or matching. Various approaches have been offered to address and cope with web service discovery challenges. In the early work, web service discovery was based on a syntactic or keyword searches which measure accordance between the user query and descriptions of service, but the appearance of new techniques such as semantic discovery, makes this technique have primarily semantic which measure the similarity between the user query and the semantic description service.

In the following, we discuss web service discovery approaches, which covers these different aspects.

**1) Syntactic-based approaches**: The idea of these approaches is very simple, where the user sends a request having keywords. These keywords are compared with descriptions of services stored on service brokers. Despite to their facility and simplicity in implementation, these approaches have some limitations because they do not present good results for the user and their software cannot test the textual descriptions prepared for human using. The approach presents a model for the similar syntactic matchmaking. This model represents a combination of interface, attributes and QoS similarities with lexical similarity. This solution aims to join all nodes of the arbitrary number either a federation or a cloud to exist in the UDDI virtual register. There is a service description part in each node. When the user sends a message requesting one of these nodes, the request moves from each node to its neighbor, and repeats this process for all nodes which get this request. The results are provided in all nodes, and then are sent to the main node.

**2)Semantic-based approaches** Approaches in this category are interested in the semantic description of services. These approaches are fast growing due to their advantages that address insufficiencies in syntactic-based approaches. In general, these approaches can be categorized into either distributed or centralized architectures. Within the centralized architectures, there are several ontologies that have been developed for this purpose. Examples are OWL S ontology and DAML-S ontology (DARPA Agent Markup Language for Services), which depend on DAML-S language. The iSeM approach, which is semantic matchmaker, that implements hybrid and adaptive semantic matching of OWL S services. The logic based of services that belong to this approach depends on the logical input/output concept subsumption relations, the accurate computation and the logical plugin relation. The SPARQLent , which is semantic matching approach that considers logic based and takes into account all functional profile of OWL S services. The SeqDisc approach, which considers a semantic approach that assess the similarity between WSs, taking into account both concrete and abstract part that form WSDL file in the similarity process.

**3) Context-based approaches:** In general, the context can be defined as information that can describe or characterize the case of an entity, where the entity can be considered as a place, a person, or an object, which can be used to interact between application and user. In the context of web services, either a user or a service may have their own contexts. The context of a service is represented as QoS, localization, cost, etc. While, the context of a user is represented as his preference, localization, etc. The approach makes use of ontology to enhance the concept of context values of user and illustrates the

relations between different context values in an automatic way. Through these context value relations, the required services are suggested to the user. The UDDI+ approach aims to apply the semantic discovery and discovers services achieving user requirements. The CASD (Context Aware Service Discovery) approach is introduced to provide a semantic web service discovery module. The approach makes use of ontologies to exploit semantic similarity, which can be used to recommend the best services corresponding to user requirement.

**QoS awareapproaches:** QoSare responsible for the quality of the provided service and is used to calculate the degree of similarity between user QoS request and QoS of provided services to achieve high degree of quality according to user needs. A two-phase approach for web service discovery depending on collaborative filtering approach and QoS is presented. The approach first applies a collaborative filtering strategy as semantic matchmaking and then makes use of QoS attributes as weights that can be given by user to get best recommended services according to user needs with correct QoS information. In WSRB (Web Service Repository Builder) approach, a web service crawler engine forwards many requests to various UDDI registries according to user requests, and then collects all QoS results. The QoS of the services, which have the same function will be represented in a matrix, and then all QoS attributes are normalized. The similarity measure or ranking score will be calculated by a weighted sum of all values of QoS attributes. The user puts the weight of each QoS attribute according to his preference. Then developed a fuzzy-based solution for the discovery process that builds a model representing the ranking of QoS-aware or non-functional web services as a fuzzy with different criteria for taking problem decision.

Some recent work has proposed annotating web services manually with additional semantic information, and then using these annotations to compose services. In our context, annotating the collection of web services is infeasible, and we rely on only the information provided in the WSDL file and the UDDI entry.

## 4.PROMINENTALGORITHMSINDISCOVERING WEB SERVICES

The prominent algorithms in discovering Web Services are

### a) Semantic Matching Algorithm-

This algorithm is used to calculate the Similarity based on the semantic information provided by the user. The overall The overall similarity is calculated by using $Similar_{des}$,$Similar_{req}$, and $Similar_{res}$:

$$\text{Similarity} = \frac{\alpha(Similar_{des}) + \beta\,(Similar_{req}) + \gamma\,(Similar_{res})}{\alpha + \beta + \gamma}$$

where $Similar_{des}$,$Similar_{req}$, and $Similar_{res}$ are the description, request, and response similarity, respectively. These components return a real value between 0 and 1, indicating the degree of similarity. The core procedure for the semantic matching algorithm is shown in Algorithm 1.

**Algorithm 1: Semantic matching**

1 for each resource S in repository
2 Compute Similardes=Sim ($Q_{des}$, $S_{des}$)
3 for each term in query request/response
4 Get clusterTerms
5 Replace term with clusterTerms
6 endfor
7 Perform pruning process
8 Compute $Similar_{req}$ and $Similar_{res}$
9 Compute Similarity
10 endfor

**b) Composable Resource Discovery Algorithm**

We describe an algorithm, which is similar to the one to support the composition of web services using semantic descriptions. The matching criteria is defined as follows: A resource S matches a query Q when all the response parameters of Q are matched by the response parameters of S, and all the request parameters of S are matched by the request parameters of Q.

In this algorithm 2, a query is matched against all resources stored in the repository. Whenever a match between a query and resources is found, it is recorded and sorted within the matches according to highest score. A match between a query and a resource consists of matching all the response parameters of the query against the response parameters of the resource; and all the request parameters of the resource against the request parameters of the query. If one of the query's responses is not matched by any of the resource's response, the match fails. Matching between requests is computed by the same process, but with the order of the queryandresource reversed.

**Algorithm 2: Composable resource discovery**

record = Ø
for each S in repository
if Matching (Q, S) then append S to record
endfor
sort(record)
Matching (Q, S) {
SemanticMatch (Q. response, S.response)
SemanticMatch (S.request, Q.request)
}

**c)QoS basedWebServiceDiscoveryAlgorithms**

One of the most prominent QoS-based WS discovery algorithm expresses each QoS-based WS description as a CSP (Communicating Sequential Processes). Then it separates the QoS-based

advertisements into two categories: the ones that satisfy completely the QoS-based request and the others that do not satisfy the request.

### 4.1 Unary Constraints Discovery Algorithm

This algorithm consists of four sequential processes a) CSP Preprocessing b) Matchmakingc) Constraint Relaxation Process d) Selection Process

**a)CSP Preprocessing:**The CSP Preprocessing process has two sequential goals: 1) minimization of offers having missing variables/metrics; 2) minimization of the number of constraints of all CSPs. The first goal is achieved by asking the WS providers, having offers whose CSP does not have unary constraints on metrics/variables that are contained in constraints at the CSP of the demand, to enrich them with corresponding constraints on these metrics/variables. The second goal is achieved by manipulating for each CSP (of QoS offers and the demand).

**b) Matchmaking:** The matchmaking process takes as input the CSPs Pi and PD of the offers and the demand and produces four types of results: super, exact, partial, fail with decreasing order of significance. Super offers not only conform to the demand but also contain better solutions. Exact offers just conform to the demand by containing a subset of the solutions of the demand. Partial offers do not conform to the demand because either they do not use some metrics of the demand or they contain (worse) solutions that are not part of the demand's solution space. So partial results are promising, especially if the first two lists of results are empty. Fail offers contain lower quality solutions with respect to the solutions requested by the demand. If only fail matches are produced, then this is an indication of an over-constrained demand. In this case, the discovery algorithm fails and an appropriate warning is issued to the WS requester.

**c) Constraint Relaxation Process:** Assume that the matchmaking process returns only partial and fail type of results. Fail match results are of no use and are not further processed. However, partial results are promising as they represent QoS offers that don't use QoS metrics of the demand or have solutions that are not included in the solution space of the demand's CSP. If the first case holds, then the solution is to find the offer(s) with the smallest set of same undefined variables/metrics and then continue to the next process to order these offers. The user gets back the ordered list and an indicating message that his query was relaxed by removing some metrics and their unary constraints. For the second case, the matchmaking process has provided three metrics for each partial offer: number of demand's violated constraints, their total weight and if offer contains better solutions that the demand. So we order the partial list of offers according first to total weight and then to the number of violated constraints. In this way, at the top will be offers having the smallest number of hard constraints and the least total weight of weak constraints. So we put these topmost offers along with their conflicting constraints at the super or exact match list according to the value of the third metric and we move to the last process in order to rank them and return them. However, the user is warned that these ordered lists represent super or exact matches only if he weakens the corresponding violated constraints list from his demand.

**d)Selection Process:** The goal of the selection process is to rank the best result lists produced by the previous processes. If super and/or exact matches exist, then they are ranked. Otherwise, there will be only partial matches to be ranked. By providing a sorted list of the best possible matches, the WS

requester is supported in choosing the best QoS offer according to his preferences, which are expressed by a list associating weights to QoS metrics.

### 4.2 Generic Discovery Algorithm

This algorithm is more generic than the previous one as it allows any kind of constraint to be used in the CSPs. However, now the notion of better or worse applied to CSP solutions is altered. The generic discovery algorithmchecks if the whole solution of the offer is worse than all solutions of the demand by assigning a preference or value to each CSP solution.

**Algorithm. [Matchmaking]** We compute the preferences $p^D_{s1}$ and $p^D_{s2}$ of the demand's CSP $P^D$ worst $s^D_1$ and best $s^D_2$ solution respectively by solving two CSOPs (minimization and maximization). For each offer's CSP $P_i$, we compute the preferences $p^i_{s1}$ and $p^i_{s2}$ of its worst $s^i_1$ and best $s^i_2$ solution respectively in the same manner as above. Then, we consider four cases:

1. If $(p^i_{s2} \leq p^D_{s1})$, then the offer is put in the fail match list.

2. If $(p^i_{s2} > p^D_{s1} \wedge p^i_{s1} < p^D_{s1})$, then the offer is put in the partial match list.

3. If $(p^i_{s1} \geq p^D_{s1} \wedge p^i_{s2} \leq p^D_{s2})$, then the offer is put in the exact match list.

4. If $((p^i_{s1} \geq p^D_{s1} \wedge p^i_{s2} > p^D_{s2}) \vee (p^i_{s1} \geq p^D_{s2}))$, then the offer is put in the super match list.

The first case expresses the fact that the offer's best solution is not better than the worst solution of the demand and justifies the classification of the offer as failed. The second case expresses the fact that the offer has some bad solutions but also some good solutions so it is considered as a partial result. The third case concerns offers that contain a subset of the solutions of the demand and justifies their classification as exact. The last case is about offers that contain not only solutions of the demand but also better ones. That's why they are classified as super results/matches. [Selection] In this process, either the best two categories of results (if not empty) or the third category are ordered based on the weighted sum of the preferences of their worst and best solutions.

## 5.TWO STAGE WEB SERVICE DISCOVERY ARCHITECTURE

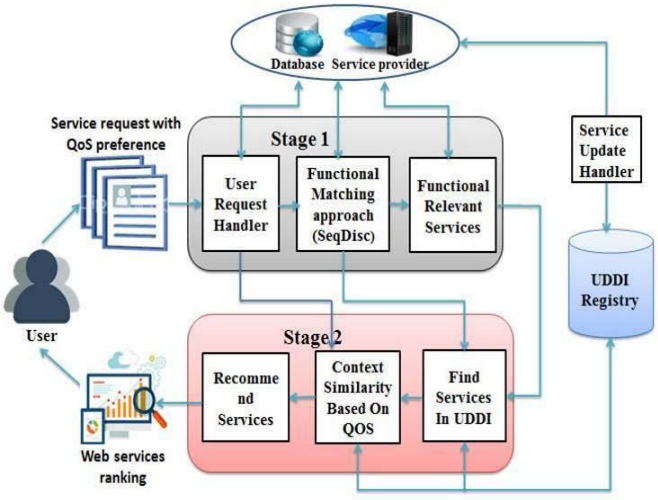


***Fig 1 Architecture of Two-Stage Web Service Discovery Framework***

In this section, we introduce a two-stage web service discovery framework, as shown in Fig 1.The proposed framework utilizes the SeqDisc approach is to filter out the available set of web services into a set that satisfy the user functional requirements (search space reduction). The framework then applies the new proposed non functional matching to get the required service (or a set of services). We discuss these two stages in details. The user request handler, found in the first stage, is used to record the information about user behavior, and then store them in profile information that will be needed in the execution stage as context-based information about the user. In the first stage, the SeqDisc approach will be used to assess the similarity between the user request as input and the other services in a UDDI registry to find the most relevant set of services to the user to achieve user functional requirements. However, this approach depends on functional parameters of the services that are represented in WSDL files, and does not consider the non-functional parameters in the discovery process. To include the contextual information in the discovery process, we propose the second stage, which considers the non-functional parameters of the services, in addition to the functional parameters that are obtained in the first stage.

## 6. CONCLUSION

An Enhanced QoS for Computing in Web Service Discovery has been presented in this paper for the purpose of finding the best available Web service during Web services discovery process based on a set of given clients QoS preferences. The use of non-functional properties for Web services significantly improves the probability of having relevant output results. The proposed solution has shown usefulness and effectiveness of incorporating QoS parameters as part of the search criteria and in distinguishing Web services from one another during the discovery process. The ability to discriminate on selecting appropriate Web services relied on the client's ability to identify appropriate QoS parameters. The proposed solution provides an effective Web service degree of similarity function that is used for ranking and finding most relevant Web services.

## REFERENCES

[1] Algergawy A., Nayak R., Siegmund N., Koppen V., and Saake G., "Combining schema and level-based matching for web service discovery," in Web Engineering, 10th International Conference, ICWE 2010, Vienna, Austria, July 5-9, 2010. Proceedings, 2010, pp. 114–128.

[2] Al-Masri E. and Mahmoud Q. H., "Qos-based discovery and ranking of web services," in Proceedings of the 16th International Conference on Computer Communications and Networks, IEEE ICCCN 2007, Turtle Bay Resort, Honolulu, Hawaii, USA, August 13-16, 2007, 2007, pp. 529–534.

[3] Bose A., Nayak R., and Bruza P., "Improving web service discovery by using semantic models," in 9th International Conference Web Information Systems Engineering, 2008, pp. 366–380.

[4] Canfora G., Penta M. D., Esposito R., and Villani M. L., "A framework for qos-aware binding and re-binding of composite web services," Journal of Systems and Software, vol. 81, no. 10, pp. 1754–1769, 2008.

[5] Chen L., Yang G., Zhang Y., and Chen Z., "Web services clustering using som based on kernel cosine similarity measure," in 2nd International Conference on Information Science and Engineering, 2010.

[6] Dong .X, Halevy A. Y., Madhavan J., Nemes E., and Zhang J., "Similarity search for web services," in The Thirtieth International Conference on Very Large Data Bases, 2004, pp. 372–383.

[7] Hao Y. and Zhang Y., "Web services discovery based on schema matching," in ACSC2007, 2007, pp. 107–113.

[8] Keidl M. and Kemper A., "A framework for context-aware adaptable web services," in Advances in Database Technology - EDBT 2004, 9th International Conference on Extending Database Technology, Heraklion, Crete, Greece, March 14-18, 2004, Proceedings, 2004, pp. 826–829.

[9] Klusch M., Kapahnke P., Schulte S., Lecue F., and Bernstein A., "Semantic web service search: A brief survey," KI, vol. 30, no. 2, pp. 139–147, 2016.

[10] Kungas P. and Matskin M., "Semantic web service composition through a p2p-based multi-agent environment," in Agents and Peer-to-Peer Computing, 4th International Workshop, AP2PC 2005, Utrecht, The Netherlands, July 25, 2005, Revised Papers, 2005, pp. 106–119.

[11] Ma .S., Chang K. Y., Lin J., Ma C., and Lin J., "Qos-aware query relaxation for service discovery with business rules," Future Generation Comp. Syst., vol. 60, pp. 1–12, 2016.

[12] Ngan L. D. and Kanagasabai R., "Semantic web service discovery: state of-the-art and research challenges," Personal and Ubiquitous Computing, vol. 17, no. 8, pp. 1741–1752, 2013.

[13] Ran S., "A model for web services discovery with qos," SIGecom Exchanges, vol. 4, no. 1, pp. 1–10, 2003.

[14] Shao L., Zhang J., Wei Y., Zhao J., Xie B., and Mei H., "Personalized qos prediction forweb services via collaborative filtering," in 2007 IEEE International Conference on Web Services (ICWS 2007), July 9-13, 2007, Salt Lake City, Utah, USA, 2007, pp. 439–446.

[15] Yan J. and Piao J., "Towards qos-based web services discovery," in Service-Oriented Computing - ICSOC 2008 Workshops, ICSOC 2008 International Workshops, Sydney, Australia, December 1st, 2008, Revised Selected Papers, 2008, pp. 200–210.