An Efficient Web Service Discovery and recommendation Framework by integrating various Web services using QoS, QoE and Service Social Network

Pratibha Pawar[1], Prof. Chavan V. G[2], Prof Gadekar P.R[3]

#ME Computer (Engineering), Dept. of Computer Science &. Engineering, Solapur University, SKN Sinhgad College of Engineering, Korti, Pandharpur, Maharashtra, India

pratpawa16 @gmail.com

***Abstract*-**Today is the world of web and services, new services mostly combine with already available services for providing a particular functionality instead of creating a new service for providing new functionalities. Though, Web services are increasingly becoming the most dominating implementation for the service oriented architecture paradigm for enterprises due to their ease of use. Their uptake on a Web scale has been significantly less than initially anticipated. One of the main causes for being the uptake significantly less is that, the isolation of services and the lack of social relationships among related services. Other reason is the lack of semantic information about the web service description at the time of publication. Hence, this paper proposes connecting the isolated service islands into a global social service network to enhance the services sociability on a global scale, considering the QoS parameters and QoE. Once, the social service network of web services is created we can use this for web service discovery as well as for web service recommendation.

Index Terms*-Web service, semantic information Global service network QoS, QoE* .

I. INTRODUCTION

Data The architecture used for designing the software is very important as it gives the early view of the software being developed. A service-oriented architecture (SOA) is an architectural pattern in computer software design in which application components provide services to other components via a communications protocol, typically over a network. Services can be combined to provide the functionality of a large software application. SOA makes it easier for software components on computers connected over a network to cooperate.

Web service is the new distributed computing paradigm for Web application, which uses SOA and is the most popular implementation of service oriented architecture. The reasons behind the web services as popular implementation of SOA are their interoperability, they use standardized protocols and have a low communication cost.

Web Services have much to offer towards interoperability of applications and integration of large scale distributed systems. To make Web services accessible to users, service providers use Web service registries to publish their services. In order to integrate these services, one must be able to locate and acquire specified services. Existing Universal Description Discovery Integration (UDDI) technology uses a central server to store information about registered Web services.

Web services use standardized industry standard protocol for the communication. All the four layers (Service Transport, XML Messaging, Service Description, and Service Discovery layers) use well-defined protocols in the web services protocol stack. This standardization of protocol stack gives the business many advantages such as a wide range of choices, reduction in the cost due to competition, and increase in the quality.

Web services use SOAP over HTTP protocol, so you can use your existing low-cost internet for implementing web services. This solution is much less costly compared to proprietary solutions like EDI/B2B. Besides SOAP over HTTP, web services can also be implemented on other reliable transport mechanisms like FTP.
Despite all the advantages of the web services, numbers of web services available on web now-a-days are much less than the expected. While there are trillions of Web pages available on the Web, the number of publicly available Web services in one large composition service system is not greater than 4000, which is very small[7]. On the other hand, a large body of research about Web service discovery has been devoted to keyword or semantic based discovery to improve the quantity and quality of service matching

performance. Nevertheless, from a recent survey[8], most services published on the Web have not been used, and only a few of the services on the Web have been discovered, composed or invoked. Lack of quality mechanism for discovering the web services affects the composition of web services as the required web services can't be discovered in an efficient manner, which in turn affects the creation and publication of the web services.

After investigating for lack in the discovery mechanisms following reasons can be inferred.
Current approaches for the web service discovery only consider services as isolated functional islands with no links to related services.
Services are considered only in terms of their own functional and nonfunctional properties and its social interaction with other peer services is ignored.

Service's sociability is the skill, tendency or property of being sociable or social, and of interacting well with related services, which is supported by the network models we refer to here as service social networks[10]. A service social network is constructed to reflect services' social reality, describe the mutual consciousness of mutual agreement about a social situation and to support the services' future social activities.

II. LITERATURE SURVEY

It proposes the approach to Considers the description of a particular web service checks for the peer services. To give weightage to link of social services this paper considers different functional and non-functional parameters of a web service. The non-functional parameter means the different QoS values. Quality of Service (QoS) has been widely used as a standard way to model and evaluate the non-functional features of a web service. Typical QoS features include reliability, response time, security, and invocation fee. QoS plays an essential role in various web service management tasks, such as selecting a service that fulfills both the functional and non-functional requirement specified by a user. It also serves as the key criterion to differentiate web services that provide similar functionality[1].

The above approach has QoS considered in it but it also has some limitations :

1. It does not always reflect in what users are interested
2. It relies on the QoS information published by service providers.

To overcome above issues this paper suggest to use the sentiment analysis of user review also to extract the feature about which the user review is talking about[2].

The paper presents an attempt to apply Neurofuzzy in the design and implementation of a rule-based scheduling algorithm to solve the shortcoming of well-known scheduling algorithms. A fuzzy-based decision maker has been proposed to compute a new priority of all CPU processes according to the process pre-priority and its execution time. Results given in this paper demonstrate that the average waiting time and the average turnaround time in the proposed algorithm are better than that obtained using pre-emptive priority scheduling, and closed to that obtained from pre-emptive shortest-job-first (SJF) scheduling. The new proposed algorithm is a dynamic scheduling algorithm which deals with both process priority and its execution time, while the pre-emptive SJF scheduling algorithm doesn't. The results obtained, using Neurofuzzy, are approximately the choice as for fuzzy but it responds faster than it. On the other hand the functional Neurofuzzy is the best algorithm compared with structural and fuzzy scheduling algorithms [3].

In [4] paper, they present how to mine product features. The proposed extraction approach is different from the previous methods because it only mines the features of the product in opinion sentences which the customers have expressed their positive or negative experiences on.

In order to find opinion sentence, a SentiWordNet-based algorithm is proposed.
There are three steps to perform our task:

1. Identifying opinion sentences in each review which is positive or negative via SentiWordNet
2. Mining product features that have been commented on by customers from opinion sentences
3. Pruning feature to remove those incorrect features.

In the [5] paper the propose below points

1. To create a synthetic dataset of WSDL/OWL-S documents, QoS parameter values and user reviews.
2. To extract features and sentiment from user comments.
3. To rank the functionally similar web services using fuzzified QoS parameters.
4. To link similar web services and form a social network of web services.
5. To compute the weight-age of each link in the social network of web services.
6. To recommend web services to user based on users query/previous usage pattern of user.
7. To create a platform that allows user to give feedback about used/searched web services.

III. METHODOLOGY

In proposed methodology, WSDL/OWLS documents are pre-processed and created a social network of services, by linking web services whose output parameters are matching with the input parameter of another web service. Each link has given two weights viz.

Functional Weight and Non-Functional Weight. Functional weight is constant and is calculated based on the relatedness of input parameters of one service to input parameter to another. While, Non-functional weight is combination of QoS and QoE. For QoS, fuzzification have used which helps in considering more than one QoS parameters at a time. And user comments sentimental analysis has been carried out to decide on QoE.

Figure 3.1 shows the overall methodology. Each individual component in the proposed methodology is described below:

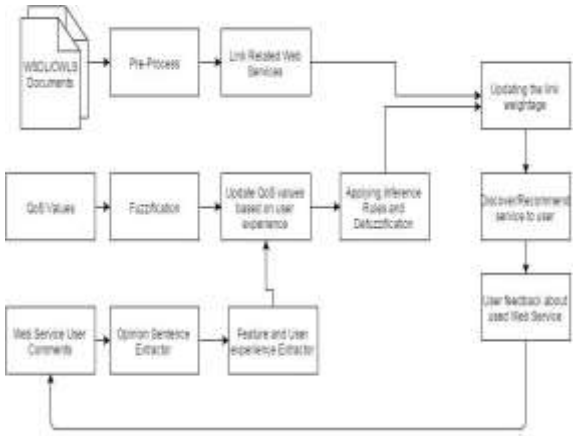

Fig 1: System Architecture

WSDL Document Preprocess: The input for this step is WSDL document and output will be a vector consisting name of web service, message names used in the WSDL, list of names of input and output parameters used in each message, description of web service etc.

Link related web services: This module will process the vector of each web service got from pre-process step and will link the related web services.

Opinion Sentence Extractor: Input to this module is user comments on all the web services whose WSDL documents are considered for forming the social network of web services. It extracts the sentences which gives the opinion about the QoS parameter of service.

Feature and User experience extractor: This module takes the opinion sentences as the input and finds the exact feature about which the user commenting and user sentiments behind it.

Fuzzification: This module takes the crisp QoS parameter values and converts those into fuzzy linguistic variable. We have used overlapping trapezoidal membership function for the same. As user gives his experience in natural language, we have to bridge a gap between crisp values and experience in natural language and hence we have used fuzzification.

Update QoS values: This module will update the QoS values based on the user experience got after processing the user comments.

Applying inference rules and Defuzzification: This module will apply the inference rules to get the resultant QoS value in linguistic variable form, which we will further defuzzify to get the resultant crisp value for a QoS parameter. We have used center of gravity mechanism for Defuzzification.

Update link weight-age: This module will calculate the weight-age of link between two web-services by using the functional and non-functional parameters along with the sentiment of user comments.

Discover/Recommend service to user: This module will Discover/Recommend the web services to user based on user query/previous usage pattern of user

3.1 Social Service Network Creation

3.1.1 WSDL/OWLS Document Preprocessing

Here, we preprocess the OWLS document to get the vector of information as mentioned above. We have used two different approaches to preprocess OWLS documents, this section explains both.

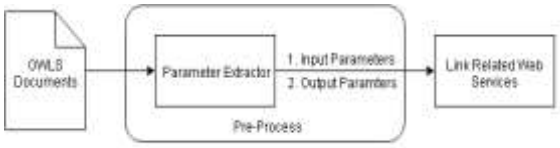

Figure 3.2: OWLS document pre-processing approach 1

In this approach, Input parameters and Output parameters are extracted from the OWLS documents. The rdf:ID attribute of <process : Input > and <process : Output> tags have been fetched to get the input parameters and output parameters respectively.

These parameters will be inputted to next module which will link the related web services based on the input and output parameters. Figure 3.2 shows the overall flow for pre-process.

3.1.2 Link Related Web Service

In this step, we linked the web services which are related to each other. We say two web services are related if the output parameters of one is matching with input parameter with other. We say, output and input parameters are matched when either they are exactly similar or output parameter is a part of input parameters. For example, consider [RECOMMENDED PRICE, BICYCLE] are output parameters of a service and [BICYCLE, PRICE, QUANTITY] are input parameters of a service. In this case output parameters RECOMMENDED PRICE and BICYCLE are matched with input parameters PRICE and BICYCLE respectively. The degree of relatedness for two web services is calculated with equation 3.1.

$$Related(WS_i, WS_j) = \frac{2 * |WS_{i,op} \cap WS_{j,ip}|}{|WS_{i,op}| + |WS_{j,ip}|} \quad (3.1)$$

where, $WS_{i,op}$ denotes the output parameters of web service WS_i and $WS_{j,ip}$ denotes the input parameters of web service WS_j .

3.2 QoE Analyzer

In opinion mining[4], users primarily care about what the customers like and dislike. So we only need to extract these sentences called Opinion sentences that people use to express a positive or negative opinion. Observing that people often express their opinions of a product feature in opinion sentences, we can extract opinion sentences from the user comments using all the opinion words. In a review, there are many sentences. Some are opinion sentences, others are irrelevant sentences. For instance, let us look at the following two sentences:

"The performance of this car is perfect."
"I have seen this car few days back."

In the first sentence, the feature, performance, is in the opinion sentence. But, the second sentence not an opinion sentence, and no feature exists. In order to find the opinion sentences, we use opinion words which are sentiment words that people used to express their positive or negative attitudes. Only four kinds of words can express the sentiment, they are nouns, adjectives, adverbs and verbs. Because we use nouns as product features,and few opinion sentences use nouns to express the sentiment. So here, we only think about adjectives, adverbs and verbs as opinion words. Sentiment includes four types mentioned in Table 3.1.

Table 3.1: Sentiment Classes and Examples

Sentiment Class	Example
Very Positive	This is a fantastic site to find any book you are looking for.
Positive	They handle the customers very well.
Neutral	Some of the books i have found overseas in UK, or other countries.
Negative	I will give low rating to this site.

Neutrality is usually used to describe a fact, and users pay much attention to the positivity and negativity. In review extraction, we take more care of the positivity and negativity of the opinion words. For evaluating the sentiment of each sentence, we make the quantization of opinion words to calculate the sentiment score of each sentence. The sentiment score of each opinion word is acquired from SentiWordNet.SentiWordNet [14] is a lexical resource for opinion mining. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity. The assumption that underlies SentiWordNet's switch from terms to synsets is that different senses of the same term may have different opinion-related properties. Each of the three scores ranges from 0.0 to 1.0, and their sum is 1.0 for each synset. This means that a synset may have nonzero scores for all the three categories, which would indicate that the corresponding terms have, in the sense indicated by the synset, each of the three opinion related properties only to a certain degree. For example, the synset, corresponding to the sense "may be computed or estimated" of the adjective estimable, has an Obj score of 1.0(and Pos and Neg scores of 0.0), while the synset [estimable(1)] corresponding to the sense "deserving of respect or high regard" has a POS score of 0.75, a Neg score of 0.0 and an
Obj score of 0.25.

From this observation, we can extract opinion sentences in the following way: for each sentence in the review database, if it satisfies that the positive or negative score is greater than certain score, to extract this sentence as opinion sentence. The positive and negative score can be calculated in below formulas. Firstly, calculate the opinion word's sentiment.

In SentiWordNet, each opinion word has different sentiment scores in different scenes. It's different to classify the opinion word into the right scene. Here, we use the sentiment
average of the opinion word in each scene as its final sentiment score in review.

$$Score_{pos}(Word_{(v/adj/adv)}) = \frac{1}{n}\sum_{i=1}^{n} Score_{pos}(i) \quad (3.2)$$

$$Score_{neg}(Word_{(v/adj/adv)}) = \frac{1}{n}\sum_{i=1}^{n} Score_{neg}(i) \quad (3.3)$$

After acquiring the sentiment of each opinion word, we need to get the adjectives, adverbs and verbs' sentiment score in the sentence respectively. The opinion word in the same part of speech has the same sentiment weight in the sentence, so we think the sentiment average in each part of speech as the sentiment score of adjectives, adverbs and verbs.

$$Score_{pos}(v/adj/adv) = \frac{1}{n}\sum_{i=1}^{n} Score_{pos}(Word_{i(v/adj/adv)}) \quad (3.4)$$

$$Score_{neg}(v/adj/adv) = \frac{1}{n}\sum_{i=1}^{n} Score_{neg}(Word_{i(v/adj/adv)}) \quad (3.5)$$

Then we calculate the positivity and negativity of each sentence as follow:

$$Score_{pos} = \frac{Score_{pos}(v.) + Score_{pos}(adj.) + Score_{pos}(adv.)}{3} \quad (3.6)$$

$$Score_{neg} = \frac{Score_{neg}(v.) + Score_{neg}(adj.) + Score_{neg}(adv.)}{3} \quad (3.7)$$

Its an opinion sentence, if Max($Score_{pos}$, $Score_{neg}$) is greater than 0.08 else its and irrelevant sentence for our project. $WSQoEScore_i$ is the QoE score of i^{th} web service. for every opinion sentence of a web service if its $Score_{pos}$ is greater than $Score_{neg}$ we add $Score_{pos}$ to $WSQoEScore_i$ else we subtract $Score_{neg}$ from $WSQoEScore_i$. This way after processing all the opinion sentence for a web service we get a final QoE score of a service.

Data: C: User Comments for Web Service, SentiWordnet
Result: Polarity of user comments
for *every comment c in C* **do**
  S: Divide comment c in sentences;
  for *every sentence s in S* **do**
    POS Tag the sentence s;
    for *every adverb a, verb v, adjective av in s* **do**
      for *every sense sc of a, v, av in SentiWordnet* **do**
        $Score_{pos}(Word_{a/v/av}) = Score_{pos}(Word_{a/v/av}) + Score_{pos}(sc)$
        $Score_{neg}(Word_{a/v/av}) = Score_{neg}(Word_{a/v/av}) + Score_{neg}(sc)$
      end
      $Score_{pos}(Word_{a/v/av}) = Score_{pos}(Word_{a/v/av})$/No. of Scenses of a/v/av;
      $Score_{neg}(Word_{a/v/av}) = Score_{neg}(Word_{a/v/av})$/No. of Scenses of a/v/av;
      $Score_{pos}(a/v/av) = Score_{pos}(a/v/av) + Score_{pos}(Word_{a/v/av})$;
      $Score_{neg}(a/v/av) = Score_{neg}(a/v/av) + Score_{neg}(Word_a)$;
    end
    $Score_{pos}(a/v/av) = Score_{pos}(a/v/av)$ / No. of (adverbs/verbs /adjective) in s;
    $Score_{neg}(a/v/av) = Score_{neg}(a/v/av)$ / No. of (adverbs/verbs /adjective) in s;
    $Score_{pos}(s) = (Score_{pos}(a)+Score_{pos}(v)+Score_{pos}(av))/3$;
    $Score_{neg}(s) = (Score_{neg}(a)+Score_{neg}(v)+Score_{neg}(av))/3$;
    if $isGreater(Score_{pos}(s), 0.08)$ **then**
      $Score_C = Score_C + Score_{pos}(s)$
    end
    if $isGreater(Score_{neg}(s), 0.08)$ **then**
      $Score_C = Score_C - Score_{neg}(s)$
    end
  end
end
return $Score_C$

Algorithm 1: QoE Analyzer Algorithm

3.3 Ranking Web Services based on QoS and QoE

3.3.1 Fuzzification

In this step, we take the crisp values as inputs like Response Time, Latency, Availability, Throughput etc., and produce as output the fuzzy sets and the degree with which the crisp values belong to these fuzzy sets using the membership functions. Membership functions are defined for a range of values of a fuzzy set and provide a membership value (from 0 to 1) for each value in that range. This membership value indicates the degree with which the given crisp value belongs to a fuzzy set. Different membership functions are available: Triangular Membership Function, Trapezoidal Membership Function, Gaussian Membership Function and others. Trapezoidal Membership function is selected out of these as it allows us to have a more number of services belong to a class with higher values as compared to triangular membership function and it requires less computation as compared to Gaussian membership function. First, the range of values of QoS parameters are noted and this range is split into 5 equal parts, with each part corresponding to a Fuzzy Set. A linguistic variable is assigned to each part, based on its lower limit and upper limit. Say for example, if the Response time range is from 1 to 100 seconds, it is split into 5 parts along with linguistic variables shown in Table3.2.

Table 3.2: Fuzzy Sets for Response Time (Non-Overlapping)

Crisp Value Ranges	Fuzzy Linguistic Variables (Fuzzy Sets)
1-20	Very Low
20-40	Low
40-60	Medium
60-80	High
80-100	Very High

Now, if we plot the graph membership degree versus Response Time, ranges belongs to x-axis and we will be having a specific degree (y-axis value) between 0 and 1 with which each Crisp value of Response time belongs to a particular fuzzy set. The ranges shown in Table 3.2 are non-overlapping and one of the disadvantage of such partitioning is it will not provide proper degree values to boundary values. For example, consider Response Time value 40 according to above ranges it belongs to Fuzzy Sets "Low" and "Medium" with degree 0 which is not proper. Hence, to provide proper justification to boundary values of these parts, it is better to have the ranges of to be overlapped. So we redefine the ranges to each part as shown in Table 3.3.

Table 3.3: Fuzzy Sets for Response Time (Overlapping)

Crisp Value Ranges	Fuzzy Linguistic Variables (Fuzzy Sets)
1-26	Very Low
14-46	Low
34-66	Medium
54-86	High
74-100	Very High

The range of parts cannot cross the extreme values 0 and 100, so they are retained as it is in the redefined range. Trapezoidal Membership functions are redefined for these new ranges of parts and each function is represented in the form of 4 points, corresponding to the points of trapezium. For a given value of Response Time, it is possible to get the membership degree of that value by plotting a vertical line from that value and finding the y-coordinate where this vertical line cuts one of the edges of a membership function. For values falling in overlapping ranges, this vertical line might produce to membership values corresponding to two fuzzy sets it belong to. Similarly the fuzzy sets corresponding to latency is also determined. These Fuzzy sets of values of response time and latency are input to the next step, Inference.

3.3.2 Inference Mechanism

This step takes as input the fuzzy sets of response time and latency and outputs fuzzy sets of new value of resultant fuzzy set for (RT,L) using Mamdani Style of Inference Mechanism[13]. To explain this style, consider the below Inference Rules Matrix:

The above Table 3.4 can be read as, if fuzzy set of response time is Very High and fuzzy set of latency is also Very High, this rule table indicates that the fuzzy set of new value of (RT,L) is Very Low. This is because a service should have low response time and low latency, But this approach indicates that the new value of fuzzy set for resultant (RT,L) belongs to Very Low fuzzy set. Using this Inference table, each fuzzy set of latency is compared with each fuzzy set of response time to obtain a list of fuzzy sets of resultant (RT,L) value. The membership degree of each of the resultant fuzzy set is the minimum of the membership degree of two fuzzy sets being compared.

Table 3.6: Inference Rules for (RT,L) and (A,TP)

(RT,L) \ (A,TP)	VH	H	M	L	VL
VH	VH	VH	H	H	M
H	VH	H	H	M	M
M	H	M	L	L	L
L	M	M	L	L	VL
VL	M	L	VL	VL	VL

Similarly, Table 3.6 can be read as, if fuzzy set of Availability is Very Low and fuzzy set of Throughput is Very Low. Then the rule table indicates that fuzzy set for new value (A,TP) is Very Low. This is because a service should have a high availability and high throughput. Table 3.6 follows the same description for the fuzzy sets of (RT,L) and (A,TP), after which will get the resultant fuzzy set for [(RT,L),(A,TP)]. Similarly, the Inference rules for [(RT,L),QOE], [(A,TP),QoE] and [[(RT,L),(A,TP)],QoE] are shown in table 3.7, which plays a major role in combining the QoS and QoE parameters together.

This list of fuzzy sets and the corresponding membership degree is the input to next step, Defuzzification.

Table 3.4: Inference Rules for Response Time and Latency

Latency \ Response Time	VH	H	M	L	VL
VH	VL	VL	VL	L	M
H	VL	L	L	M	M
M	L	L	L	M	H
L	M	M	H	H	VH
VL	M	H	H	VH	VH

Table 3.5: Inference Rules for Availability and Throughput

Availability \ Throughput	VH	H	M	L	VL
VH	VH	VH	H	H	M
H	VH	H	H	M	M
M	H	M	L	L	L
L	M	M	L	L	VL
VL	M	L	VL	VL	VL

3.3.3 Defuzzification

Defuzzification is the last step in Fuzzy Inference System. This step takes as input the fuzzy sets of resultant (R,TL) or (A,TP) or [(R,TL),(A,TP)] value depending upon the users choice of QoS parameters and generates new value in the crisp format. Different Defuzzification methods are available, namely Maximum Membership principle, Center of Gravity technique, Weighted Average method and many others.Center of Gravity Defuzzification method is the most accurate of all and hence chosen for this problem.

Table 3.7: Inference Rules for (RT,L)/(A,TP)/[(RT,L)(A,TP)] and QoE

(RT,L)/(A,TP)/[(RT,L)(A,TP)] \ QoE	VH	H	M	L	VL
VH	VH	VH	H	H	M
H	VH	H	H	M	M
M	H	M	L	L	L
L	M	M	L	L	VL
VL	M	L	VL	VL	VL

3.4 Updating Link Weightage

For every link in WS SSN has two types of weightage related to it those are:

• Functional Relatedness Weight: This weight refers to how related output parameters of one WS are to the another web service, which is calculated by equation 3.1. This weight remains same for every link irrespective of the user query.

• Non-Functional Weight: This weight refers to QoS parameters and QoE. As, for every query user might have different QoS parameter preferences, and based on those QoS parameters, the set of services satisfying user need may change. Based on these changed set of services, every time the fuzzified values of QoS parameters and QoE will changed, which results in a change in final combined weight of QoS and QoE. And hence, for every link in SSN this weight varies according to user preferences.

The total weight of the link is defined as the addition of both the weights. As, the Non-Functional weight of a link is varying with each user query we have to update the link weightage before start of discovering the appropriate web services for user.

3.5 Discovering the Web Services

At this step, we have the user query and WS SSN. Based on the user query we search for the services such that the output parameters of the services matches with the user query. If a service with exactly matched output parameters is not found then we try to composite two different web services which can together satisfy the user need. For composing the web services, we start with services whose output parameters matches more with the user query, then we search for its neighboring services in SSN whose output parameters matches with the remaining user query. After getting all such services we sort them on the basis of link weightage to get the final composite services.

IV. RESULT AND ANALYSIS

4.1 Experimental Setup

The defined methodology is implemented in the Core Java 1.8 programming language. Java is chosen for implementation cause, it provides an efficient set of data structure through its Collections framework. Also, cause of platform independence of Java. Experiment has been carried with a synthetic dataset explained above. Algorithms were implemented using Java v1.8. The experiment was conducted on a machine with the Intel core i7 processor, 8.00GHz, running Windows 7.

4.2 Dataset

We created a synthetic dataset which consisting of WSDL and OWLS documents, QoS values and User comments for each WSDL/OWLS documents. We have used user review comments. So we mapped each WSDL document to particular set of QoS values in dataset. For sentiment analysis, we crawled http://www.sitejabber.com/ to get the user comments for each WSDL document. The number of comments per WSDL document varies from 7 to 989.

4.3 Result Analysis

4.3.1 User Interface

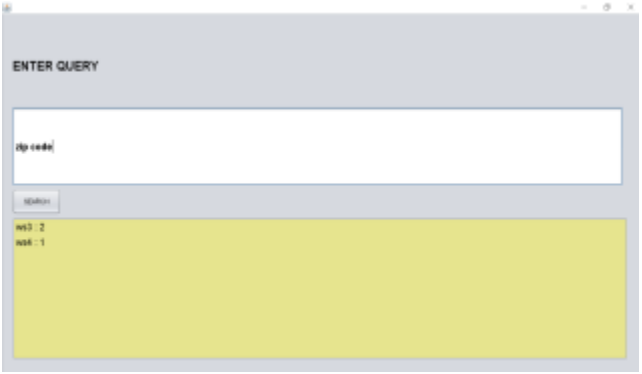

Fig 4.1 UI Interface

In the user interface user query has to be entered and he will be given list of services applicable to his input along with the service associated or sociable with this service.We store the values for non-functional properties in the database Different options available for QoS parameters(columns) are viz. Response Time and Latency, Availability and Throughput. The user gets the services according the database values updated as per user comments choose different priority values for every QoS parameter are Very Low, Low, Medium, High, Very High.

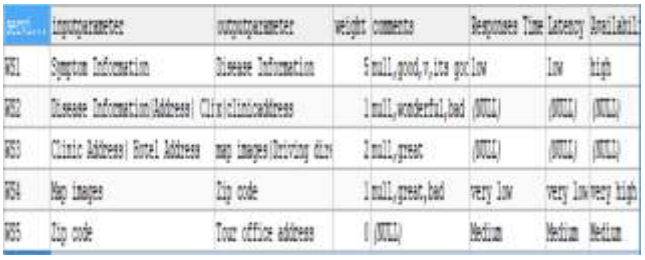

Fig 4.2 Binding table simulator of WSDL document (Database table with QoS and QoE)

Fig 4.3 UI Interface(user comment form)

User can put his review comments on the respective service which will be used put positive or negative score of the service and ultimately used to increase the ranking.

4.3.2 WSDL/OWLS Pre-processing

The extracted input and output parameters using Approach 1 is shown in Figure 4.1,where each line shows OWLS file name, list of input parameters followed by list of output parameters.

```
ws1==symptominformation==diseaseinformation==5
ws2==diseaseinformation|address|cliniccity==3|clinicaddress==1
ws3==clinicaddress|hoteladdress==mapimages|drivingdirections==2
ws4==mapimages==zipcode==1
Match: ws4
inputparameter: mapimages
select servicename,weight from registered_services where LOWER(REPLACE(outputparameter, ' ', '' )) like '%mapimages%'
servicename1: ws3
servicename1: ws3
weight1: 2
ws5==zipcode==tourofficeaddress==0
```

Fig 4.4 Output of pre-processing

4.3.3 Web Service Social Network

Figure 4.5 shows part of service social network formed. The actual SSN has 1000 nodes each representing a WS and total of 41254 edges. Each edge has two scores the first is functional relatedness weight (on first line of every link in Figure 4.5) and second is Non functional weight (on second line of every link in Figure 4.5). Every link is directional from one WS to another. In Figure 4.5, we can see that their are web services (for example WS3 and WS 122) which are not connected to any other web services called as isolated web services and they can not be combined with any other web services.

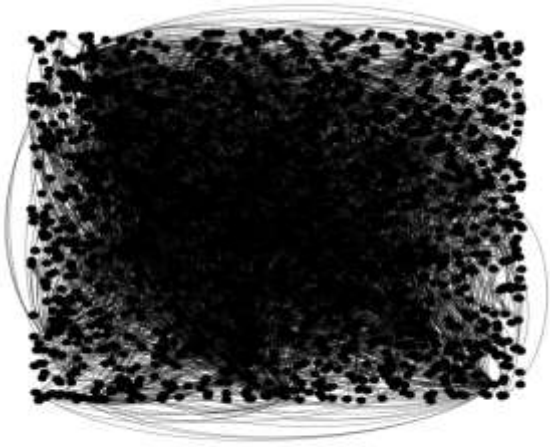

Figure 4.5: Web Service Social Network

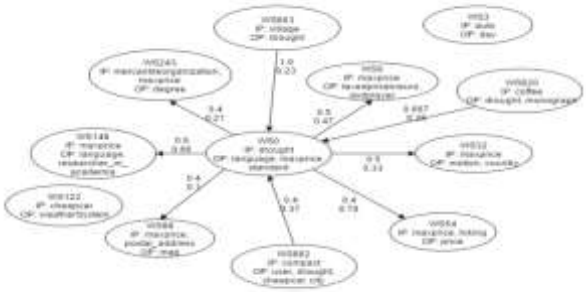

Figure 4.6: Web Service Social Network in detail

V. CONCLUSION

The Current UDDI structure is such that a web services is not having any knowledge about the other web services.Due to this there is limited scope for the web services Discovery But, as per this paper web service can know about other web services in some way. Then user can have a better web service discovery. option as when he queries for a specific web service, all the related services can be detected and can be provided to the user improving the user experience. This paper uses the concept of social networking to create a links between the web services using many different functional, non-functional parameters along with the user comments about the web services.

REFERENCES

1. Wuhui Chen, Incheon Paik, Patrick C.K. Hung, "Constructing a Global Social Service Network for Better Quality of Web Service Discovery", IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 8, NO. 2, MARCH-APRIL 2015
2. Xumin Liu, Arpeet Kale, JavedWasani, Chen Ding, and Qi Yu, "Extracting, Ranking, and Evaluating

Quality Features of Web Services through User Review Sentiment Analysis", 2015 IEEE International Conference on Web Services
3. ShathaJawad, “Design and Evaluation of Neurofuzzy CPU Scheduling Algorithm”, Networking, Sensing and Control (ICNSC), 2014 IEEE 11th International Conference.
4. Weishu Hu, Zhiguo Gong, JingzhiGuo, "Mining Product Features from Online Re-views", IEEE International Conference on E-Business Engineering 2010
5. E. Al-Masri and Q.H. Mahmoud, “InvestigatingWeb Services on the World Wide Web," in Proc. 17th IEEE Int'l World Wide Web Conf., 2008, pp. 795-804.
6. W. Jiang, D. Lee, and S. Hu, “Large-Scale Longitudinal Analysis of SOAP-Based and RESTful Web Services," in Proc. 19th IEEE Int'l Web Serv. Conf., 2012, pp. 218-225.
7. Shatha Jawad, “Design and Evaluation of Neurofuzzy CPU Scheduling Algorithm”,
8. Networking, Sensing and Control (ICNSC), 2014 IEEE 11th International Conference.
9. Z. Maamar, H. Hacid, and M.N. Huhns, “Why Web Services Need Social Networks," Proc. IEEE Internet Comput., vol. 15, no. 2, pp. 90-94, Mar./Apr. 2011.
10. Weishu Hu, Zhiguo Gong, Jingzhi Guo, ”Mining Product Features from Online Reviews”, IEEE International Conference on E-Business Engineering 2010
11. F.Y. Wang, D. Zeng, K.M. Carley, and W. Mao, \Social Computing: from Social In-formatics to Social Intelligence," IEEE Intell. Syst., vol. 22, no. 2, pp. 79-83, Mar./Apr. 2007
12. Yen, J., Langari, R.: Fuzzy Logic; Intelligence, Control, and Information. Prentice Hall, Englewood Cli s (1999)
13. W. Jiang, D. Lee, and S. Hu, “Large-Scale Longitudinal Analysis of SOAP-Based and
14. RESTful Web Services,” in Proc. 19th IEEE Int’l Web Serv. Conf., 2012, pp. 218-225.
15. J. Davies, J. Domingue, C. Pedrinaci, D. Fensel, R. Gonzalez Cabero, M. Potter, M.ichardson, and S. Stincic, “Towards the Open Service Web,” BT Technology Journal, vol. 26, no. 2, 2009.
16. C. Petrie, “Practical Web Services,” IEEE Internet Computing,vol. 13, no. 6, pp. 94–96, 2009.