# Retrieving Real Time Web Services through Web Scrapping

*Dr. Sumathi,
Information Science & Engineering, Canara Engineering College, BenjanaPadavu, Bantwal,Karnataka, India

Dr. Niranjan N Chiplunkar,
NMAMIT, Nitte, Karkala, Karnataka, India

Mrs. Sadhana B,
Information Science & Engineering, Canara Engineering College, BenjanaPadavu, Bantwal,Karnataka, India

Mr. Vasanth Nayak,
Information Science & Engineering, Canara Engineering College, BenjanaPadavu, Bantwal,Karnataka, India

*Abstract:* **This paper focused on retrieving Web service links through Web scrapping. Request for Web service's names will be given as user input to the proposed system. Web scrapping method retrieves the Web Service Definition Language links of the Web services and these links' operation names are compared with requested user input. If Web services found suitable, then such Web services will be clustered and invoked. If requested Web services are not available then WSDL links of synonym services retrieved and stored in Synonym Matching Clusters.**

Keywords: Web Scrapping, Web Service, WSDL, Clusters

## I. INTRODUCTION

Web scrapping is method of retrieving the data from Web and extracting suitable and required information. Web services are remote services which provide set of functions during their remote execution and returns results. WSDL is Web Service Definition Language which describes Web Services in XML form and shows all operation names with their input, output elements, data types and schemas.

The proposed research uses search engine to search the requested functional word as Web services using query as "functional_word?WSDL". This query returns the cluster of WSDL of Web services which are semantically similar to the functional word and are called as "community Web services". The cluster gives the WSDL which are having semantically similar operations and more than one WSDL links are retrieved for a requested functional word. But among these results, some of the functions may not be matching with the user requested functions. Therefore, it is necessary to process each of the WSDL to find suitable Web services. Java based APIs are used to process WSDL and to invoke Web services. Methodology with block diagram represents how to retrieve suitable WSDL links through Web Scrapping.

The organization of this paper is as follows. Section II is focused on literature review, section III is on Web Scrapping, section IV is focused on results and section V is focused on conclusion and future work.

## II. LITERATURE SURVEY

The detailed literature survey is discussed the existing methods of Web service selection and composition. The proposed system selects suitable Web services through WSDL processing and clustering. If user requested input

parameters are not available in the selected Web services, then composition/integration of Web services is carried out. Therefore literature review on Web service selection and composition is carried out in detail.

[1]In the previous work of Web Service discovery, it is focused on dynamic discovery of Web services using Web and WSDL processing.

[2]In earlier research work of WSDL weight matrix calculation, it is focused on selecting suitable Web Services according to highest calculated weight of many similar Web Services.

[3] Selection of semantically similar Web services is carried out with the help of Ontologies and Mining the QoS parameters of many functionally similar Web services are based on user's preference which uses the ranking algorithm. Authors take user requirement as inputs and considered user feedback and QoS parameters to rank the Web services.

[4] Selection of the best user desired service without user interaction is proposed using a novel concept called skyline. The two important concepts defined by service model are service schema and service model. Composite Web services are represented by service graph which includes set of operations that need to be followed in the sequence to achieve the goal. Then they perform topological sort on this graph, which orders the operations based on their dependencies. Service Execution plans are dynamically generated for each request. Since the Web Service operations are static, the QoS factors of each operation is stored in the repository and indexed by operation IDs. Execution plans which are dominated by other plans are filtered according to the QoS factors.

[5] jUDDI which is pronounced as "Judy" is a Java implementation of UDDI and it is open source specification for UDDI. jUDDIv3 services supported by JDK 1.6 or later. jUDDI includes extensive predefined tModels like Quality of Service Metrics. This is built on JPA standardized interfaces, and tested with Apache OpenJPA and Hibernate.

Because of absence of UDDI registry, the proposed research do not use UDDI. Therefore only way of testing Web service QoS metrics is by executing the Web services.

[6]WS-Policy provides the conditions for providing the service. A consumer sees these conditions and decides whether to use this service or not. The basic assertions like *<wsp:exactlyone></wsp:exactlyone>, <wsp:all></wsp:all> indicate set of policy assertions.*

- The WS-Poilcy assertions can be inserted inside input , output, fault, operation, port, or in binding WSDL elements.
- It can be inserted as child elements of them, or else it is referred using *<wsp:PolicyReference>* element.

[7] WSDL analyzer is a free tool available to check the correctness of the WSDL and also to give information about each element of WSDL. This system takes WSDL as input and returns all the elements and their sub-elements. Even though WSDL analyzer validates and returns the WSDL elements, the proposed system cannot extract these elements from the browser and it is cumbersome to extract those elements from browser and to use it. Therefore, proposed research extracts WSDL elements only from WSDLHelper API and can use it to invoke Web services.

[8] XLANG is the language for describing the services' behavior. For a Web service XLANG enhances WSDL language by addition of behavior element. Behavior defines two types of actions: WSDL operations and timeout operations. The contract element of XLANG is also used for inter connection of several XLANG service descriptions. The context element supports the transactions, which lead to cancel or rollback the transactions in case of failure. Conversion of XLANG data type to BPEL4WS data type is given by the standards specified.

[9]In the published paper of earlier version of this work, it is focused on full implementation of WSDL processing & invoking the Web Sevices using WSDLHelper class.

## III. WEB SCRAPPING

Web scrapping is nothing but extracting require information from search results of Web. Here Bingo search engine is used to search for required Web Services and when a query is given as Query_word?WSDL, result returned will be matched WSDL files & additional links with details..

From these results, to extract only WSDL links Jsoup API is used. JSoup.Connect method is used to extract required results from search results.

The statements to retrieve the links are given below.

```
Document dc = Jsoup.connect ("http://www.bing.com/search? q="+Query_word+"%3Fwsdl").get();
Elements lnk = dc.getElementsByTag("a");
//get links present in HTML page
                for(Element lnk : links)
                {       String Hreflink = lnk.attr("href");
                        URL.add(Hreflink);
                }
```

As a result of the search query, the Bingo search engine retrieves web result that has WSDL links as well as additional information in the web browser. From this additional information, it is require extracting only WSDL links.

The output of this Code snippet is extracted WSDL files.

The count of these links gives retrieved number of links. The WSDL files retrieved here are processed to extract <service> element. This element contains exact <service> name offered by the perticular Web service.

Methods for processing WSDL definition is stored in javax.wsdl.Definition class.

```
Definition def=w.getDefinition(dfile)
```

The above statement creates the object of the definition class. The service element is to be stored in the hash map structure. Therefore Map sampleMap2 = new HashMap() creates the hash map.

```
sampleMap2 = def.getAllServices();
```

This statement stores all service elements in the hashMap named sampleMap2. Then creating the object of QName class and storing the key part of hashMap in the serviceName is done by

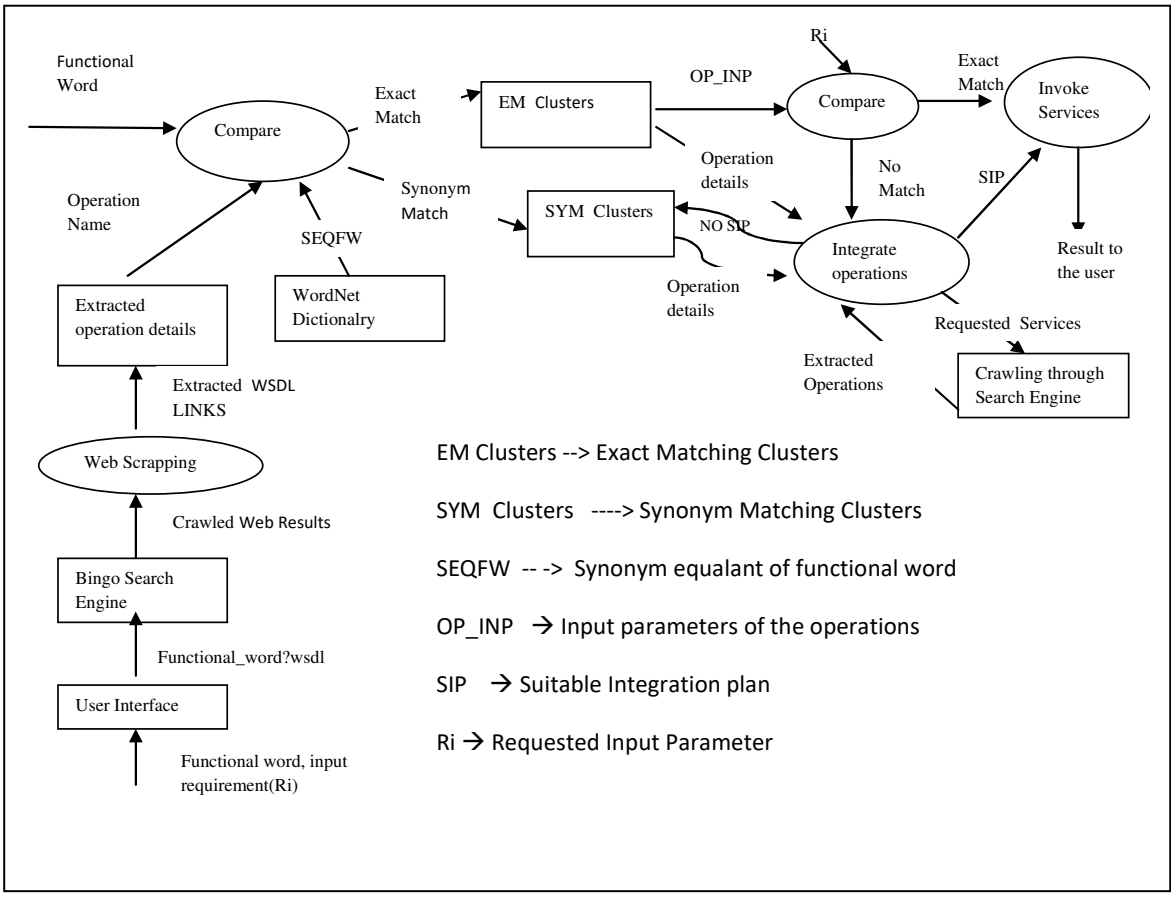
serviceName=qName.getLocalPart(). Then finding the pattern match between given functional word and <service> name is performed by using pattern match with regular expression.

The regular expression of the pattern match facilitates the required matching and gives the count of matched URLs of the functional word with <service> element. The next step is to apply equation (5.1) to find search precision or support of the given search.

$$\text{Support} = \frac{\text{Count of Matched Service Name to functional word}}{\text{Count of Retrieved WSDL}} \quad (1)$$

If support is>0 then those WSDL records are processed once again to get information about operation names, input/output parameters, pre-condition, effect elements etc. Here it is observed that to extract this information from the WSDL, two methods can be followed. The first method is storing all the required WSDL in the local repository in different files. The second method is retrieving each WSDL from online and overwriting it on a local file and processing these WSDL files. The first method requires huge storage to store all these WSDL in different local repository. The second method consumes internet bandwidth to download each WSDL to process. The procedure is given in the figure 3.1.

Figure 3.1 Data flow diagram of Web Scrapping



The functional word of the user request is inputted to the proposed system. A non-zero value of search results' precisions (confidence and support) are used to find whether the functional words are available as Web service or not? If the requested terms are available as Web services, then those service's WSDL records are retrieved and interpreted to select suitable Web services.

The functional word given by the user requests may exists as single Web service or as composed/integrated results of available online Web services depends on matching of input parameters of these Web services to the user given inputs.

Given a user request Sr = <F, Ri> contains functional word F and requested input Ri. The set OW contains retrieved WSDL links of available Web services for user requested functional word in the World Wide Web. Let OW=<wsdl1, wsdl2, wsdl3….wsdln> and n is an element of the finite set of natural numbers. The Web service is selected if F exists as operation name of any of the WSDL of the set OW and the input Ri may or may not matches to any input parameter of these operations.

- EM Clusters : Let CS = <op1,op2..opn> are the operations which are matching exactly to the given functional word and CS is cluster of exact matching operations.
- SYM Clusters: If operation names of retrieved operations are synonymially matching to requested functional words then cluster formed from such operations are called as SYM Clusters.

**3.1 Integrate operations process :** Integrating operations required when input parameters Si=<in1,…inn> of these clusters are not matching to user given input Ri. Therefore Ri ∉ Si. Then input elements of these operations are to be extracted and should be searched as <operation> / <output> of existing available Web services. Let INi= <in1, in2….inn> are the extracted input parameters of $i^{th}$ operation of the set CS where i ranges from 1 to n and n is an element of the finite set of natural numbers. Now a search for these parameters as <operation> / <output> of the existing online Web services and feeding it as the input of the operation of set CS is the process of integration of the Web services.

**3.2 Crawling through search engine:** Searching and Web scrapping operations are called to crawl through Web for searching the required Web Services.

## IV. RESULT

Table-1 Result of Web Scrapping

| Requested Word | Extracted Service Name | WSDL Links after Web Scrapping |
|---|---|---|
| Temperature | GlobalWeather | http://www.Webservicex.com/globalweather.asmx?WSDL |
| | TemperatureConversions | http://Webservices.daehosting.com/services/TemperatureConversions.wso?WSDL |
| | WeatherForecast | http://www.Webservicex.net/WeatherForecast.asmx?WSDL |
| | CurrencyConvertor | http://www.Webservicex.net/CurrencyConvertor.asmx?WSDL |
| London Gold Fix | LondonGoldAndSilverFix | http://www.Webservicex.net/LondonGoldFix.asmx?WSDL |
| GeoIP | GeoIPService | http://www.Webservicex.net/geoipservice.asmx?WSDL |

Table I shows the result of Web Scrapping when different functional words given as input to the Web search engine. When user input is given with keyword "Temperature" then four different WSDL links are extracted from Web Scrapping process. Extracted Service names are given in the table 4.1, which shows availability of those services. Similarly for other requests results are shown in the table.

## V. ANALYSIS

As per the results of WOOGLE[10], it provides higher precision and lower recall. By considering parameter matching and concept matching, WOOGLE obtains the recall as high as CONONLY and precision as high as PARONLY only. WOOGLE shows top- 5 precision. The top-2, top-5, and top-10 precisions of WOOGLE are 98%, 83%, 68% respectively which are higher than those of the two naive methods by 10 to 30 percentage points.
But Woogle aquires Web service descriptions through UDDI registries. In this system search is performed through keywords, and offers the possibility to discriminate between inputs and outputs. The proposed system does not depend on UDDI because its absence. The proposed system uses only Bingo search engine to acquire WSDL.

[11] VUSE is another example of specialized Web Service Search Engine that retrieves the WSDL. VUSE also depends on UDDI to retrieve WSDL and it retrieves total eight WSDL links for "weather" as search query. But now a days all the WSDL links do not contain the executable services. Therefore our system invokes the services and based on the availability of the services it generates the composition plan. Even though VUSE retrieves more links than our system, VUSE depends on UDDI to retrieve the WSDL links. Many links among these are not having real Web services or they are out of service. But our system does not depend on UDDI, instead it uses search engine. Therefore it is justified that our system is better compared to VUSE system.

[12] WESS (Web Service Search Engine), uses a Web Ontology Language for Services - OWL-S, Semantic Annotations for WSDL is SAWSDL and also non-semantic (Web Service Definition Language - WSDL). The architecture of the WESS consists of Web crawler, the WSDL/SAWSDL and OWL-S parsers. The exchange of information is performed through WSDL, OWL-S or XML custom files.

## VI. CONCLUSIONS & FUTURE WORK

This research have not used any simulation tool, instead this work implemented using JDK & Eclipse which are open access. This is one of advantage of implementation of this work. This research performs real time search from Web which produces dynamic results for current available Web Services. As a future work this research is extended with [13]implementation of integration of dynamically available Web services according to user requirement.

## REFERENCES

[1]Sumathi, Niranjan N. Chiplunkar, Ashok Kumar A., Dynamic Discovery of Web Services using WSDL. In I.J. Information Technology and Computer Science, 2014, 10, 56-62, 10.5815/ijitcs.2014.10.08
[2] Pawar, S. and Chiplunkar, N.N. 'Populating parameters of web services by automatic composition using search precision and WSDL weight matrix', Int. J. Computational Science and Engineering, Vol. 17, No. 3,pp. 333 - 342
[3] Divya Sachan, Saurabh Kumar Dixit, Sandeep Kumar, "A System for Web Service Selection Based on QoS", International Conference on Information Systems and Computer Networks, IIT Roorkey, 2013 pp 139-144.
[4] R.Joseph Manoj, Dr.A.Chandrasekar, "A Literature Review on Trust Management in Web Services Access Control", International Journal on Web Service Computing (IJWSC), Vol.4, No.3, September 2013, pp 1-18.
[5] http://juddi.apache.org, The Apache Software Foundation, Licensed under the Apache License, Version 2.0 Copyright © 2014.
[6] Web Services Policy 1.5 – Framework http://www.w3.org/TR/ws-policy/
[7] Online WSDL Analyzer, https://www.wsdl-analyzer.com/service/service/600982223? version=1.
[8] Microsoft. Web Services for Business Process Design (XLANG), http://xml.coverpages.org/xlang.html.
[9] Dong, X., Halevy, A., Madhavan, J., Nemes, E., Zhang. J., "Similarity Search for Web services" , Proceedings of VLDB, 2004, pp 372-383.
[10] Ourania Hatzi, Georgios Batistatos, Mara Nikolaidou, Dimosthenis Anagnostopoulos, "A Specialized Search Engine for Web Service Discovery", International conference on Web Services Computing IEEE- 2012.
[11] Chen, Wuhui, Incheon Paik. "Improving efficiency of service discovery using Linked data-based service publication." Information Systems Frontiers 15.4 (2013): 613-625.
[12]Pawar, S., Chiplunkar, N.N.: Open source APIs for processing the XML result of web services. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1848–1854. IEEE, September 2017.
[13]Sumathi Pawar, Dr. Niranjan N Chiplunkar, Dynamic Composition of Web Services by Service Invocation Dynamically, I.J Education and Management Engineering(IJEME), 2017, 4, pp 41-50.