Cloud based Mobile Web Service Discovery using Semantic Matching and Key-Word based Approach

Samson D. Plavila
Computer Engineering
Thadomal Shahani Engineering College
Bandra, India

Ruhi Bajaj
Asst. Prof. Computer Engineering Department
Thadomal Shahani Engineering College
Bandra, India

ABSTRACT

Use of web-services in devices with low processing capability has increased with the advancement in wireless technologies. Web-services are mainly used to reduce overhead of heavy processing in resource limited environment. Many public web-services are available that can be used to retrieve information or perform a task .Web-Service discovery is the process of discovering relevant web-services for the user based on users specification. Discovery of relevant web-services are mainly based on two methods text based matching and semantic matching. Searching for web-service is resource consuming and low-end devices or mobile devices cannot efficiently perform the task. A cloud-server can be an excellent candidate for this job as its resources are not limited and can be scaled easily they are cost effective. In this paper a frame-work for discovering relevant mobile web-services using a cloud based web-service matching algorithm is proposed. The frame-work is expected to provide efficient results and quick response time.

Keywords

Cloud server, service discovery, semantic matching.

1. INTRODUCTION

There are many methods proposed for discovering web-services most of them are based on either text based matching or semantic matching. Text based matching though easier may not always fetch all the relevant mobile web-services since there might be web-services that are semantically similar, on the other hand semantic based matching using ontology is time consuming and ontology file needs to be generated for reasoning. In the proposed frame-work matching of web-services by combing key-word based matching method and wu and palmer[3] similarity method which calculates similarity by calculating depths of two concepts in wordnet[2] database. The description of web-service will be in hREST[1] description format and a parser will be used to retrieve required elements from the file. To increase the speed of searching algorithm the web-service descriptions are clustered based on different categories they belong .Thus matching algorithm won’t be applied to all web-service descriptions and only to those belonging to relevant category which is also determined using a matching algorithm. The discovery algorithm and web-service description repository will be stored on cloud-server and will be accessed by mobile device through rest web-service. On receiving list of mobile web-services user can test the web-service on the mobile device itself from the interface created dynamically using the hREST[1] web-service description file.

2. RELATED WORK

Many researches and frame-works have been proposed to overcome drawbacks of web-service discovery and increase efficiency of result. Each of them focus on different issues, most of them are still at an early stage.

The frame-work suggested in [7] is for web-service discovery based on semantic matching using ontology in the environment of Multi-Agent the method doesn’t use cloud-server.
The work done in [8] proposes a proxy based web-service discovery approach where a web-server act as middle man and does the processing part this method uses SOAP method which has an overhead of compressing and decompressing message

The frame-work proposed in [4] is similar to approach proposed in this paper but the algorithm used for matching is different they use ontology reasoned for semantic matching in contrast wordnet[2] database is used for semantic matching and also performing text-based matching for better result.

3. DESCRIPTION

The figure 1 shows the different processes in the frame-work and its flow. For the purpose of simplicity the framework description can be divided into different modules each module will describe the part it plays in the frame-work the modules are as follows.

User Interface

1. User Query Processor
2. Service Parser and Reader
3. Category Key-words file
4. Service Matchmaking
5. Ranking of Web-Service.

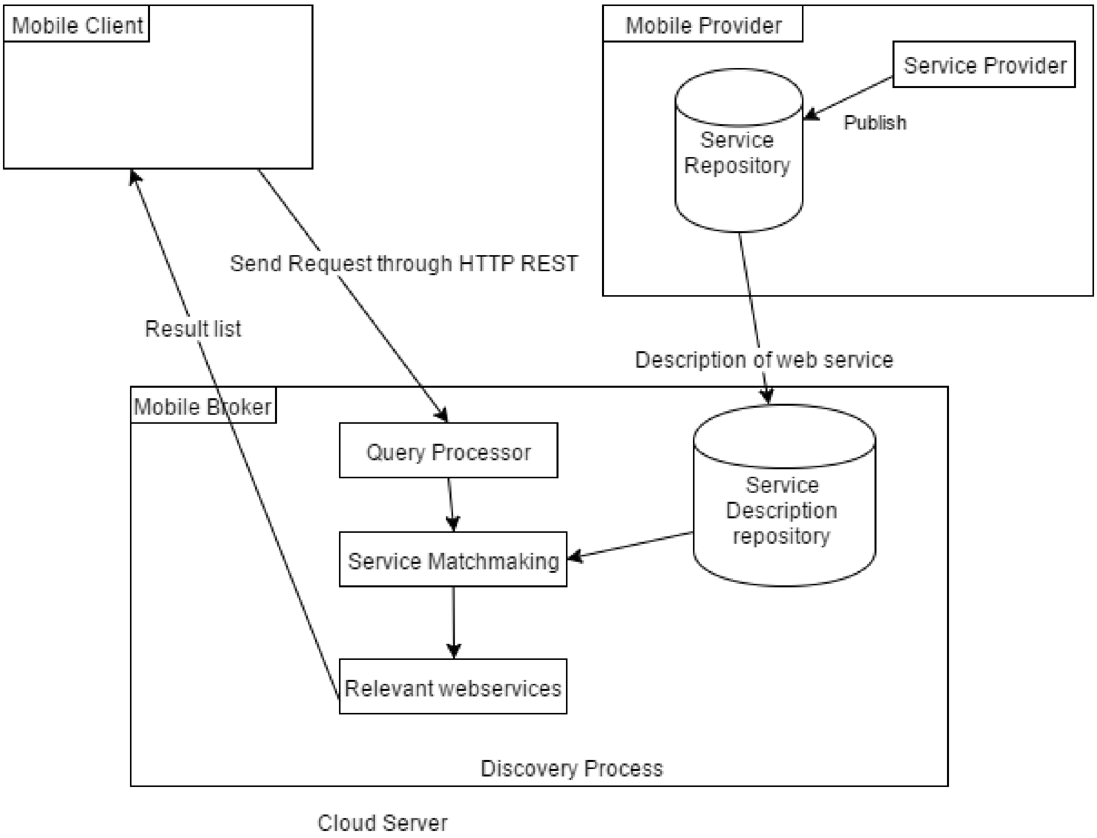


Fig 1:Mobile Web-Service Discovery Framework

3.1 User Interface

This is the client part of the frame-work a user will type description of web-service it requires in a few words. This description will be sent to cloud-server using REST web-service where discovery of web-service is done using match-making algorithm and a list of relevant web-services is generated which is sent back to client as response. The client can also test the web-services using the interface which is dynamically created using the web-service description received from the web-service list.

3.2 User Query Processor

The description received from user is in plain text it needs to be processed before it is sent for matching query processing involves following steps:

1. Stop word removal

   - Some words in the query text are very common and including them won't help much in discovery process so the words that are common such as "and, is, are, an…" needs to be removed, so first step of query processing is removing the stop-words.

2. Word Stemming

   - Word stemming is the process of reducing word to its root word
     For e.g.: word "description" after stemming will be reduced to "describe".

3. Tokenization

   - Tokenization is the process of extracting words or tokens from a sentence, tokens are extracted from the user query by splitting based in spaces.

3.3 Service Parser and Reader

A Service reader is used to extract information from the hREST[1] web-service description file and parsing is the process of retrieving required elements from the file such as input and output parameters, web-service description etc there are many libraries available for reading xml files .JSOUP parser will be used for processing hREST[1] web-service description file.

3.4 Category Key-words file

This process is not included in discovery algorithm it is used to generate a file which consists of all the keywords from the web-service description files of web-services separate by category. This file will be used later for classifying the category to which the user query send by client will belong.

3.5 Service Matchmaking

This is the step where matching of the web-service description with the user query is done the process can be described in following steps:

1) Matching Algorithms:

   a) Based on the keywords obtained after query processing, matching is performed both keyword based and semantic and average of both keyword and semantic score is taken to determine the category of query.

   b) Keyword based matching is done by checking no. of key-words matched with the description key-words.

   c) Formula:
      i) Score S=M/N
      ii) Where

(1) M= No. of key-words matched
(2) N= total no. of key-words

d) The score obtained will always be between 0<S<1.

e) Semantic matching is done using wordNet[2] database and applying Wu & Palmer .The Wu & Palmer measure (wup)[3] calculates similarity by considering the depths of the two concepts in the UMLS, along with the depth of the LCS .

f) Formula:

i) Each key-word is matched with key-words extracted from web-service description to obtain similarity matrix.

For eg:

$$\textbf{sMatrix=} \quad \begin{array}{c c} & \begin{array}{cc} T1 & T2 \end{array} \\ \begin{array}{c} S1 \\ S2 \end{array} & \begin{bmatrix} 0.68 & 0.98 \\ 0.86 & 0.56 \end{bmatrix} \end{array}$$

Where S1, S2 are key-words of set S and T1, T2 are key-words of set T. The values in matrix indicate similarity score between respective key-words

The similarity values from the matrix are used to calculate combined similarity value of key-words using the following formula [5].

Matching average:

```
int m = sMatrix.length;

int n = sMatrix[0].length;

double simScore = 0.0F;

double maxS_i = 0.0F;

double sumS_i = 0.0F;

for (int i = 0; i < m; i++) {

        maxS_i = 0.0F;

        for (int j = 0; j < n; j++) {

                if (maxS_i < sMatrix[i][j])
        maxS_i = sMatrix[i][j];

        sumS_i += maxS_i;

                        }

simScore = sumS_i / (float)m ;
```

This similarity is computed by dividing the sum of similarity scores of all match candidates of both sentences S and T by the total number of set tokens. The value obtained is based on each of the individual similarity values, thus the overall similarity always reflects the influence of all the key-words.

2) Category classification
a) Text based matching is performed between the tokens obtained from user query and key-words in the category key-words file using the above method for text matching
b) Semantic matching is done between the tokens obtained from user query and category name using above formula for semantic matching.
Formula

CategoryScore(C) = (T + S)/2

T: Text Similarity Score

S: Semantic similarity score

3) Web-Service Matching:
a) After getting category the keywords are matched with description and input output parameters of web-services in that category

b) Ranking of web-service is based on average of three scores.

i) score =C+K+S/3
Where:

C is category score.

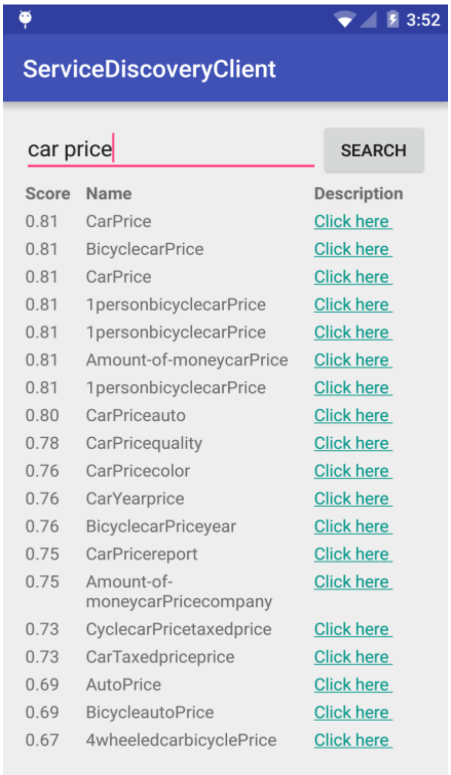
K is textual similarity score.

S is semantic similarity score.

4. IMPLEMENTATION

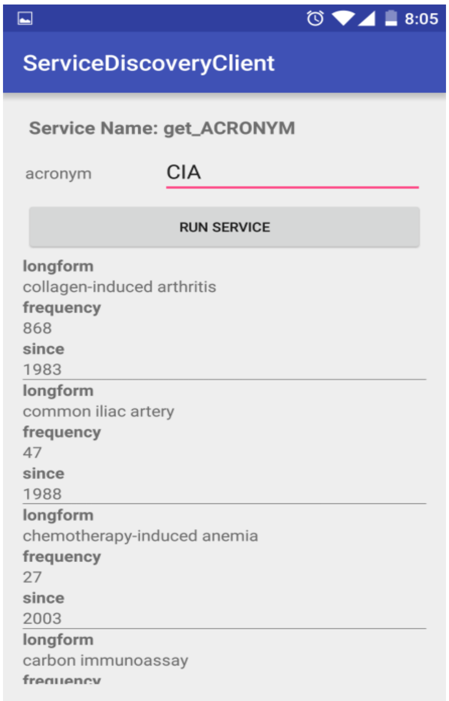
4.1 User Client Interface

An android device will be used for mobile client interface for querying web service. Web service query will be done using REST protocol. Figure 2 shows the interface for client device where user will enter search query and will get web-service list as response Figure 3 shows test interface for web-service.



Score	Name	Description
0.81	CarPrice	Click here
0.81	BicyclecarPrice	Click here
0.81	CarPrice	Click here
0.81	1personbicyclecarPrice	Click here
0.81	1personbicyclecarPrice	Click here
0.81	Amount-of-moneycarPrice	Click here
0.81	1personbicyclecarPrice	Click here
0.80	CarPriceauto	Click here
0.78	CarPricequality	Click here
0.76	CarPricecolor	Click here
0.76	CarYearprice	Click here
0.76	BicyclecarPriceyear	Click here
0.75	CarPricereport	Click here
0.75	Amount-of-moneycarPricecompany	Click here
0.73	CyclecarPricetaxedprice	Click here
0.73	CarTaxedpriceprice	Click here
0.69	AutoPrice	Click here
0.69	BicycleautoPrice	Click here
0.67	4wheeledcarbicyclePrice	Click here

Fig 2: User Interface



Service Name: get_ACRONYM

acronym CIA

longform
collagen-induced arthritis
frequency
868
since
1983

longform
common iliac artery
frequency
47
since
1988

longform
chemotherapy-induced anemia
frequency
27
since
2003

longform
carbon immunoassay
frequency

Fig 3: Web-service test interface

4.2 Cloud Server

The web-service discovery service was deployed on Amazon EC2 cloud server. On Amazon EC2, AWS Elastic Beanstalk service with Java as application platform and tomcat 8 as server was used to host the web-service which performed the task of matchmaking and ranking of web-services.

4.3 Data used

Experiment where conducted with hREST-TC3 [1] test collection [1] which consists of 1088 services. These web-services are from different categories such as food, economy, travel etc.

4.4 Programming Language and libraries

Implementation of the project was done in java, w4js library was used for wordnet support and JSOUP parser was used to parse xhtml files and extract required information.

Standford Natural Language Processing library was used to perform language processing on text.

5. DISCUSSION

In this section the implemented frame-work will be analyzed its effectiveness and how it will be suitable for emerging mobile platforms the discussion will be based on following parameters:

- Effectiveness.
- Speed.
- Suitability to platform.

Table 1.Query execution result

Keyword-word used	**No. of relevant web-services returned**	**Rank of returned web-services (out of 100)**	**No. of web-services matched**
Movie information	12	66	418
recommended price for drinks	2	60-57	392
get address from longitude and latitude	23	48-38	347
mp3 player price	20	67-53	418
Retail store service	3	50-45	418
population density of city	1	68	347
get address from longitude and latitude	23	48-38	347

5.1 Effectiveness

The effectiveness of a web-service can be measured by how relevant the results are to user .The table 1 above shows some of the queries fired for analysis which showed that the web-services returned are most relevant when there is both text based and semantic similarity in user query with the web-service description. Second where the web-service descriptions with only text similarity. The web-services with only semantic similarity showed less score and where relevant in some cases.

5.2 Speed

The queries took 5-6s for response on average which can be reduced further by increasing resources of cloud-server current configuration is AWS 't1.Micro' instance which has 613MB RAM single core processor.

5.3 Suitability to recent platforms

In this section applicability of frame-work to emerging mobile platforms such Android, Iphone, Microsoft…etc is discussed. Since all the processing is carried on cloud-server any platform can easily integrate the framework by calling web-service discovery API and passing the user Query and display the response in their required format only requirement is to support REST protocol to make API calls.

6. CONCLUSION

Mobile devices are becoming most significant interface for mobile computing environment. Significant advancement in wireless technology and internet speed has helped in accessing information anywhere at any time which has resulted in increase in web-services. Discovering relevant web-services is becoming more and more important. A cloud-based service discovery method was proposed for discovering relevant web-services by semantic and key-word based matchmaking. A preliminary proto-type was developed using Android device as base for mobile device and Amazon Elastic beanstalk for cloud server. The proposed frame-work efficiently discovers relevant web-services with quick response time by shifting all the heavy processing work to cloud-server. The matching process still needs tuning as in some cases desired result might not be obtained. The future work will include improving efficiency of search result and improve ranking based on user feed-back.

7. REFERENCES

[1] http://projects.semwebcentral.org/projects/hrests-tc/

[2] Nor Azizah Saadon, Radziah Mohamad," Cloud-based Mobile Web Service DiscoveryFramework with Semantic Matchmaking Approach" 2014 8th Malaysian Software Engineering Conference (MySEC)

[3] WordNet is available at http://wordnet.princeton.edu.

[4] Perea-Ortega, José M., M. A. García-Cumbreras, and L. A. Ureña-López, "Applying NLP techniques for query reformulation to information retrieval with geographical references," Emerg.Trends Knowl. Discov. Data Mining, Springer Berlin Heidelb.,pp. 57–69, 2013.

[5] Wu & Palmer Words Similarity, "http://blog.thedigitalgroup.com/sagarg/2015/06/10/words-similarityrelatedness-using-wupalmer-algorithm/"

[6] Semantic similarity measurement," http://www.codeproject.com/Articles/11835/WordNet-based-semantic-similarity-measurement"

[7] Hang Wu and Chaozhen Guo "The Research and Implementation of Web Service Classification and Discovery Based on Semantic", Proceedings of the 2011 15th International Conference on Computer Supported Cooperative Work in Design

[8] Hassan Artail, Kassem Fawaz, and Ali Ghandour"A Proxy-Based Architecture for Dynamic Discovery and Invocation of Web Services from Mobile Devices", IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 5, NO. 1, JANUARY-MARCH 2012