International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161
©2014 INPRESSCO®, All Rights Reserved
Available at http://inpressco.com/category/ijcet

General Article

Web Service Search Engine

Rathod Amit Kashinath[Ȧ*] and Bhole G.P.[Ȧ]

[Ȧ]Department of Computer Engineering, Veermata Jijabai Technological Institute, Mumbai, India

Accepted 28 May 2014, Available online 01 June 2014, **Vol.4, No.3 (June 2014)**

Abstract

An increasing number of Web services are created and composed to construct complex business processes. Selecting an appropriate service from a lot of independently developed services which have the same functionality but different QoS properties is essential for the effective use of the composite service. However, the public UDDI Business Registry, the primary service discovery mechanism over the Internet has been shut down permanently and existing approaches heavily rely on the information in WSDL (Web Service Descriptive Language) files, which is quite limited to understand the web service's functionality and QOS. The limitation of information weakens the effectiveness of existing approaches. In this paper we proposed web search engine on the basis of WSDL and WADL (Web Application Descriptive Language) file descriptions, Web service related description on Internet, Web service reputation by feedback of user and QoS of Web service. We use this information for effective to choose the best of web services.

Keywords: *Web service search engine; WSDL; WADL; QoS; SOA; REST*

Introduction

Web of services. In order to enable efficient Web Service discovery on large scale we need to access a large number of publicly available services (Zeina Azmeh et al, 2011). The public UDDI Business Registry the primary service discovery mechanism over the Internet has been shut down. So the presently users are facing the problem for searching the web services (Chen Wu et al, 2008).

The Web services are of two types SOA and REST (Representational State Transfer), for giving information of web service they provide WSDL and WADL respectively. Presently most of the system searching based on the WSDL and WADL file which is quite limited to understand the web services QOS and Reputation (Zeina Azmeh et al, 2011; Meng Li et al, 2011).

The application software profit growth is driven by products based on Service-Oriented Architecture (SOA). Moreover, with the rising evolution of SaaS (Software-as-a-Service), increasing software or even hardware will fit into the SOA paradigm (Chen Wu et al, 2008). Various service users need to know what kind of services is available on the Web, their detailed capabilities and how they can be used under different business contexts. This has made service discovery of paramount importance. Service discovery paves the way for conducting further important SOA activities such as service binding, sharing, reusing, and composing in a dynamically changing business environment.

The RESTful web services are becoming an alternative to traditional SOAP-based web services, and their use is expected to overtake that of the SOAP-based web services (Yong-Ju Lee et al, 2011). The growing number of RESTful web services available on the web raises the challenging issue of how to locate the desired web services.

In this paper we proposed web search engine on the basis of WSDL and WADL file descriptions, Web service related description on Internet, Web service reputation by feedback of user and QoS of Web service. We use this information for ranking the Web service

Related Work

Meng Li, Junfeng Zhao, Lijie Wang, Sibo Cai and Bing Xie give the CoWS An Internet-Enriched and Quality-Aware Web Services Search Engine in which they give the SOA related web service search engine on the basis of QOS, Reputation and web description (Meng Li et al,2011). Whereas Chen Wu and Elizabeth Chang give the crawler method for web service on the basis of WSDL file which is helpful for SOA related web service (Chen Wu et al, 2008). In these both the approach they consider SOA related web services.So we propose the web service architecture by combining above two approaches and adding RESTFul web service in it. By combining the RESTFul with SOA this architecture is helpful for searching public cloud service which is made in REST or SOA as well as for searching RESTFul web services (Talal H. Noor et al, 2013).

System Architecture

In This paper we propose web service search architecture

*Corresponding author: **Rathod Amit Kashinath**

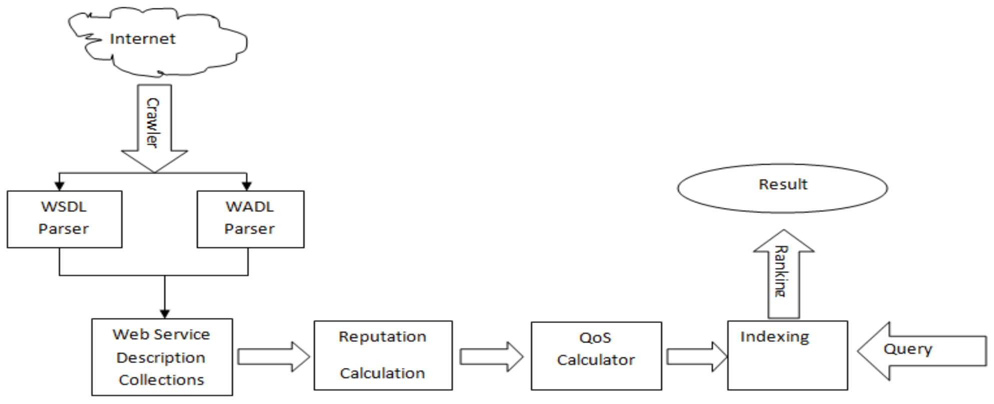

Fig. 1 System Architecture

for searching the web service. This architecture use the three main points for searching the web services which can reduces all limitation of exiting web service searching like limited description, not considering performance and choosing the best web service from many web service which has same functionality. These three points are as follow.

A. Consideration of reputation of web services

The reputation is the important factor in web serviced search engine. The reputation is calculating the feedback from current user of web service in the form of their experience (Meng Li et al, 2011).

B. Considering HTML file description related to web service

The WSDL and WADL file are in XML formats they give the information related to web service's description, method, input formats and output formats which are quite limited (Meng Li et al,2011). So we use the web description which is given in the form of HTML format on their website.

C. Comparing web service on the basis of QOS

On the internet many web services are present which provide the same functionality. So no proper comparison factor is present. We use the QoS of web service on that basic we compare same functional web services. So in fig 1 we propose the architecture contain WSDL and WADL Crawler, Web Service description collector, Reputation Calculator and QOS Calculator

WSDL AND WADL crawler

Crawling is an essential function for a Web search Engine as it provides primary input for indexing and Searching. In general, web crawlers utilize the graph structure of the Web to move from one page to another. When thinking of applying Web crawling techniques into WSDL and WADL, several characteristics distinguish the crawling algorithm. First, there are no links in WSDL and WADL files that can connect one WSDL and WADL file with another. Aiming at describing a single Web service, WSDL and WADL is not meant to be used for connecting different Web services. Hence, the crawler cannot simply generate more WSDL and WADL pages out of several seeds WSDL files. This problem has made crawling WSDL/WADL files become difficult and different from existing crawling approaches relying on 'graph search problem' analysis (Chen Wu et al, 2008; Talal H. Noor et al, 2013; Gautam Pant et al).

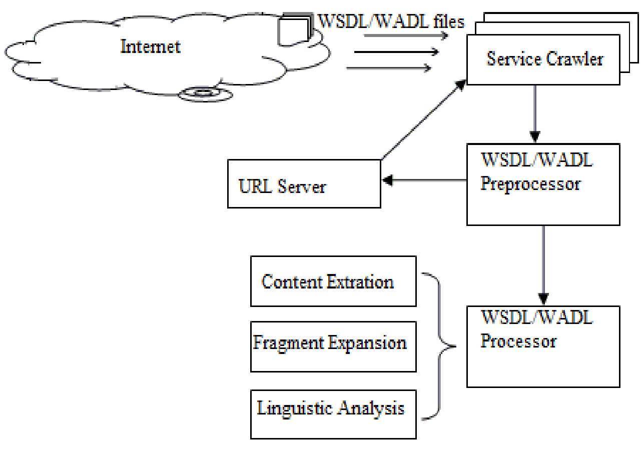

Fig. 2 WSDL and WADL Crawler

Fig 2 gives the WADL and WSDL crawler which goes through URL to URL and collects all WSDL/WADL files by using the URL server and WSDL/WADL preprocessor. The crawler uses the URL which is provided by URL server. The crawler goes to that URL and collects all WSDL/WADL file. The WSDL / WADL preprocessor used for validating the WSDL and WADL file. In this Each WSDL and WADL file is parsed by a regular WSDL parser and WADL parser respectively. When encountering invalid WSDL/WADL files, the algorithm checks whether it appears to be a HTML file. If yes, it is then sent to an HTML parser in attempt to find out some HTML links that might point to 'real' WSDL/WADL files. These potential WSDL/WADL links are sent to *URLServer*, waiting for the next round of crawling (Chen Wu et al, 2008). This collected data are send to the WSDL/WADL processor which contain three parts.

A. Content Extraction

In content Extraction the WSDL/WADL file extracted and collect the data. The WSDL /WADL file are in form xml which is self-descriptive in form. In these file they give the information of method name their input type and their output formats which is helpful for understanding the web service. For example the WSDL file contain the following node

- **Types**– a container for data type definitions using some type system (such as XSD).
- **Message**– an abstract, typed definition of the data being communicated.
- **Operation**– an abstract description of an action supported by the service.
- **Port Type**–an abstract set of operations supported by one or more endpoints.
- **Binding**– a concrete protocol and data format specification for a particular port type.
- **Port**– a single endpoint defined as a combination of a binding and a network address.
- **Service**– a collection of related endpoints.

In WADL file contains resources, method, param, option, request and representation. Each WADL specification contains at least one <resources> element. The <resources> element contains one attribute, base, that defines the domain and the base path of the endpoint, The <param> Specifies parameters used for requests to this resource, The <option> specifies possible set of values for the parameter, the <method> defines a method associated with this resource, The <request> Specifies one or more <param> elements that are passed to the method and the <representation> Specifies the media type and payload passed in the body of the request. This information captures from this WADL/WSDL file and collects data this data is used for understanding the web services.

B. Fragment Expansion

It is a process during which plain texts extracted from XML nodes are manipulated in order to improve the retrieval precision. The basic idea of fragment expansion is to retain some important structure-related information before XML markup is removed to produce plain texts (Chen Wu et al, 2008). We define as follows two primary objectives of *fragment expansion* (FE) for WSDL/WADL-based service retrieval.

First, FE shall provide different weights to term occurrences at different WSDL/WADL locations. For example, “nmtokens” appearing in some WSDL/WADL elements are generally more significant than others in expressing the overall capability of a Web service (Chen Wu et al, 2008). Consider two Web services A and B. Suppose A can provide comprehensive weather information (e.g. temperature, humidity, wind, waves, trend, climate comparison, etc.) for geographic regions all over the world. Service B, on the other hand, focuses on delivering local tourism information such as hotels, flights, restaurants, banks, and weather forecasts. It is clear that Web service A is more relevant than B in response to service requests such as ‘weather report’. However, this cannot be guaranteed since the frequency of the word ‘weather’ in Service A might be exactly the same as in Service B. Second, FE shall retain some important structural information to cater for service retrieval at various levels of granularity. It is evident that converting all token names from WSDL/WADL elements into plain texts will eliminate all structural information, which sometimes determines the service relevance to a user query. This is particular the case when searching the WSDL/WADL element “<operation />”, where words in an operation name constitute a complete independent business meaning (Chen Wu et al, 2008).

In order to achieve these two objectives, we have utilized three techniques introduced in – *Reduplication*, *Reordering*, and *Addition*. *Reduplication* is useful to achieve the first objective of FE – i.e. to provide different term weights based on its WSDL/WADL locations. Example in WSDL tokens in elements such as “<service />” and “<port type />” are generally more important than others in describing the overall capability of a Web service. Therefore, these tokens are reduplicated in order to increase their relative weights by doubling the raw frequency of tokens in important elements. Since the whole tokens will be duplicated, duplication will not break the original term sequence in name tokens (Chen Wu et al, 2008).

C. Linguistic Analysis

The Extracted node token cannot use directly they need to be converted to natural languages before being indexed using IR models (Chen Wu et al, 2008).

For example, the operation name “GetLastTradePrice” shall be tokenized into four sequential terms – “Get”, “Last”, “Trade”, and “Price”. This tokenization is down easily by using upper case letter in operation and separate that word .but this method is not possible in this case such as “GetMyeBayServices”, or “downloadMP3Music”. For example, applying the simple capital letter rule for the first token gives the result “Get”, “Mye”, “Bay”, and “Services”. This is not desirable since the company name “eBay” is mistakenly split into two different terms. As a result, service retrieval on “eBay” cannot match this operation.

So we use the tokenization method which is *Maximum Matching Algorithm* (MMA), (Chen Wu et al, 2008; J.-S. Chang et al, 1997). the basic idea of MMA is to use an external word list (e.g. a Chinese dictionary) to verify the possible word tokens parsed out from the unsegmented text. The algorithm starts from the first character in a text and reads in one character at a time to form the ‘character sequence’ ***cs***. After reading each character, it attempts to find in the word list the longest word ***w*** that starts with the current character sequence ***cs***. If ***w*** can be also found in the text (i.e. the remaining part of ***w*** also matches the following characters read from the text), the MMA marks a boundary at the end of ***w*** and starts again from the following character using the same longest word matching strategy until reach the end of the character sequence. If none matching words can be found in the word list, the first character in the character sequence ***cs*** itself will be identified as a single word.

Web service description collections

In this we collect the related webpage for web service and calculate the similarity between that webpage and web services by using distance and text similarity. For that we collect web page using WSDL/WADL URL, Service Name and End Point by providing to Google API. That collected web page is parsed and collects the all text data by removing all HTML tags that called Description Text (DT). This DT is used for calculating the relevance between web service and DT by using formula (Meng Li et al, 2011).

$$Rel_{i,j} = \frac{Sim_{i,j}}{\log_{10}(DT_{i,j}+1)+1}$$

Where $Rel_{i,j}$ is the relevance between the ith web service and the jth DT; $DT_{i,j}$ is the distance from the DT to the WSDL URL; $Sim_{i,j}$ is the DT's textual similarity to the information extracted from the WSDL file.

Reputation calculation

The Reputation of web service is calculated by using feedback from users or their experience of the user. These feedbacks are in the form of subjective which is in two form rating and comment. First, we calculate reputation using ratings and comments respectively; second, the reputation values are combined into one. The reputation is calculated and normalized using ratings as follows (Meng Li et al, 2011).

$$R_{rating\ i} = \frac{\sum_{j=1}^{m} R_j}{m*Range}$$

Where *Rj* is an end user's rating of a Web service on a website; *m* is the total number of times the service has been rated; *Range* is the range provided by the website, e.g. the range is1~5 for ratings on Service-Finder.
For the comment we use the sentiment analysis. In this we extract positive word and negative word. In this Positive word contains *good*, *great*, *perfect*, *cool*, and *excellent* where as in negative word contain *bad, hard*, *worst*, *wrong*, *useless*, *error*, etc. by using this we assign the 1 for positive comment , -1 for negative comment and 0 for neutral comment and calculate the reputation by using comment as follows (Meng Li et al, 2011) .

$$R_{comment\ i} = \frac{\sum_{j=1}^{m} R_j}{m}$$

Where *Rj* is value of a comment after sentiment analysis; *m* is the total number of comments on a Web service After getting these two reputations we calculate final reputation by combining these two reputations as follow

$$R_i = \frac{R_{rating_i}+R_{comment\ i}}{2}$$

qos calculation

QOS (Quality Of Service) is the measure of the degree to which a Web service can perform the required functionality for one or more given parameters. It is required in comparing web service which has same functionality or same methods. In this system we use the QOS for searching the best of two or many web service which has the same functionality or ranking the web service based on QOS property (Kyriakos Kritikos et al, 2009). In this system we use the three QOS property for ranking the Web Service.

A. Response Time

It is the time taken by the web service for giving the response for their request. It is the working time or it indicate the how fast is web service. This Response Time is calculated by using following formula (Kyriakos Kritikos et al, 2007; Eyhab Al-Masri et al, 2009).

$$\mathrm{QOS_{rt}} = \mathrm{RT_c} - \mathrm{RT_r}$$

Where RTc represents the time a response is received and RTr represents the time the actual request has taken place

B. Availability

Availability is defined as the ratio of the time period when the web service is available or ready for use (Kyriakos Kritikos et al, 2007; Eyhab Al-Masri et al, 2009) and it is the time for which web service is available for use. It is calculated by following formula

$$QOS_{av} = \frac{uptime}{total\ time}$$

Where uptime represents the time the system exists during the interval of time when the measurement is taking place, and the total Time represents the addition of the uptime and the downtime. The downtime represents the time the system has been unavailable during the interval of time when the measurement is taking place.

C. Throughpu

It is the maximum number of requests that can be handled or processed at a given unit time. It is calculated by following formula. (Kyriakos Kritikos et al, 2007; Eyhab Al-Masri et al, 2009).

$$\mathrm{QOS_{tp}} = \frac{\mathrm{Req_c}}{\mathrm{T_s}}$$

Where $\mathrm{Req_c}$ represents the total number of requests completed, and $\mathrm{T_s}$ represents the unit time

Conclusion and Future Work

In this paper, we propose the solution for problem in web service search engine. By using the related web page description for searching the web services which help in searching and understanding the web service functionality, the reputation for considering top popular web services and the QOS for comparing same functional web service so that we solve the limitation of WSDL/WADL file, give alternative to UDDI and comparison by the QOS which help in choose best web service. Service-oriented computing (SOC) and Web services are believed to be one of the most important enabling technologies for cloud computing. So In the future we can use this for public cloud searching

References

Zeina Azmeh, Maha Driss, Fady Hamoui, Marianne Huchard, Naouel Moha and Chouki Tibermacine (2011) Selection of Composable Web Services Driven by User Requirements, *in IEEE International Conference on Web Services.*

Chen Wu and Elizabeth Chang," Searching services (2008), on the Web: A public Web services discovery approach, *in Third International IEEE Conference on Signal-Image Technologies and Internet-Based System.*

Meng Li, Junfeng Zhao, Lijie Wang, Sibo Cai and Bing Xie(2011), CoWS: An Internet-Enriched and Quality-Aware Web Services Search Engine, *in IEEE International Conference on Web Services*

Yong-Ju Lee and Chang-Su Kim (2011), A Learning Ontology Method for RESTful Semantic Web Services, *in IEEE International Conference on Web Services.*

Kyriakos Kritikos and Dimitris Plexousakis(2007), Requirements for QoS-Based Web Service Description and Discovery, *in 31st Annual International Computer Software and Applications Conference.*

Eyhab Al-Masri and Qusay H. Mahmoud (2009),Understanding Web Service Discovery Goals, *IEEE International Conference on Systems, Man, and Cybernetics San Antonio*, TX, USA.

Talal H. Noor, Quan Z. Sheng, Abdullah Alfazi, Anne H.H. Ngu and Jeriel Law(2013),CSCE: A Crawler Engine for Cloud Services Discovery on the WorldWideWeb, *20th International Conference on Web Services.*

Kyriakos Kritikos and Dimitris Plexousakis (2009), Requirements for QoS-Based Web Service Description and Discovery,*IEEE Transaction on Services Computing*, VOL. 2, No. 4.

Gautam Pant, Padmini Srinivasan and Filippo Menczer, Crawling the Web ,[online] available: http://goo.gl/kT637L.

J.-S. Chang and K.-Y. Su (1997), An Unsupervised Iterative Method for Chinese New Lexicon Extraction, *International Journal of Computational Linguistics*, Chinese Language Processing.