International Journal of Database Theory and Application
Vol.9, No.3 (2016), pp.37-46
http://dx.doi.org/10.14257/ijdta.2016.9.3.05

An Optimal Agent Based (Oab) Architecture for Web Service Discovery

Suganya. D[1], Revathy.A[2], R.G. Suresh Kumar[3], N. Moganarangan[4] and D.Madhavan[5]
[1, 2, 5] *Master of Technology, Department of Computer Science and Engineering, Rajiv Gandhi College of Engineering & Technology*
[3] *Research Scholar, Vels University, Pallavaram, Chennai.*
[4]*Research Scholar, Rajiv Gandhi College of Engineering and Technology Kirumampakkam, Puducherry.*
[1] *vidu0902@gmail.com,* [2] *revathyaruna@gmail.com,* [3] *aargeek@gmail.com,* [4] *rengannsj77@gmail.com,*
[5] *madhavan.d21992@gmail.com*

Abstract

Service Discovery is done based on Keyword Match, Simple Semantic Description Services, and Rich Semantic Description Service. Web Services raise the web to a new level by integrating computational aspects. Web Services are accessed by computer programs but these still need help by humans. Web-services should be described in a formal, semantic way, so programs can find, compose and invoke them automatically. Existing system discover the service based on QoS (Quality of Service). It uses the Agent to discover the service, but normalization & classification are not done by Agent here. Existing system are using the selection algorithm, which works in association with QoS value, based on consumer requirements. So In our proposed system we proposed the Normalized, Classification, & Ranking of the QoS parameter based on minimization maximization criteria using agent based architecture. We rank the services according to their QoS levels and their services matching.

Keywords*: QoS; normalization; classification; ranking*

1. Introduction

Web service is deployed on the web by service providers, and their functions are described in a service description implemented in WSDL document or a set of WSDL documents. This is a registry of web services, a functions provided by the web services, and protocols used by them. The functions of the components are: service provider: creates the web service and it service definitions, and publishes the service in the service registry based on the standard UDDI (universal description, discovery and integration) specification.

This UDDI offers a set of API's to ask the service registry for the needed web services. Web services used primarily as a resource for business to communicate with each other and with clients, Web services allow organizations to impart data without familiar knowledge of each other's IT systems behind the firewall. Web services do not require the use of browsers or HTML. Web services are sometimes called application services. Web services are about the integration of applications via the web.

The simple Web services platform is XML + HTTP. The HTTP protocol is the most performed on Internet protocol. XML provides a language that can be performed between various platforms and programming languages and calm expresses complex messages and functions.

ISSN: 2005-4270 IJDTA

1.1. Web Service Agent Architecture

The basic three components of architecture are service provider, service requester and service registry. A service provider is responsible for building a useful service, creating a service description for it, publishing that service description to one or more service registries and receiving service invocation messages from one or more service requestors.

A service requestor is responsible for finding service description published to one or more service registries, and for using service description to bind to or invoke services hosted by service providers. The service registry is responsible for advertising service descriptions published to it by service providers and for allowing service requestors to search the related object of service descriptions contained within the service registry.

Publish- The publish operation is an act of service registration or service advertisement. It acts as the contract between the service registry and service provider.

Find- With the find operation, the service requestor states one or more search criteria such as type of service, Quality of Service (QoS) and so on. The result of the find operation is a list of service description that matches the find criteria.

Bind- The bind operation embodies the client-server relationship between the service requestor and service provider. The Bind operation can be dynamic such as on the fly generation of a client side proxy based on the service description used to invoke the service.

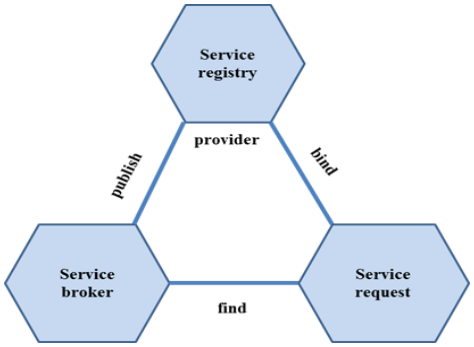

Figure 1.1. Architecture of Web Service Agent

1.2. Motivation

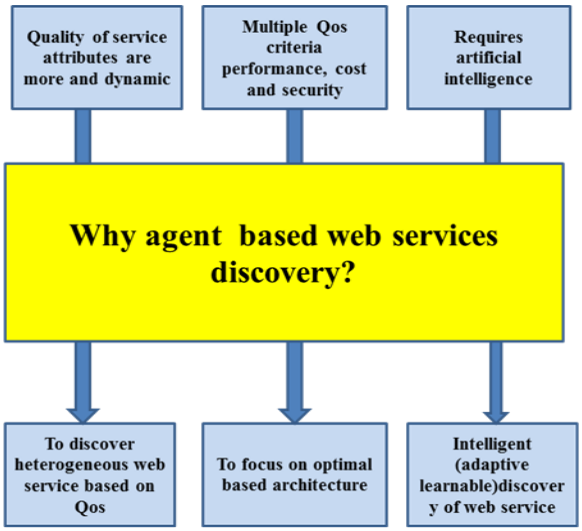

Figure 1.2. Motivation of Agent Based Web Service

2. Related Work

2.1. Web Service Discovery

Discovery of Web services is a primary principal of research in ubiquitous computing. Many researchers have focused on discovering Web services through a centralized UDDI registry. Although consolidated registries can provide effective methods for the discovery of Web services, they undergo from problems associated with having centralized systems such as single point of termination, and bottlenecks. In addition, other issues relating to the scalability of data mimicked, providing notifications to all subscribers when performing any system upgrades, and handling versioning of services from the same provider have passionately motivated for researchers to find other alternatives.

METEOR-S provides a discovery conversion for publishing Web services over federated registries but this solution does not provide the means for effective advanced search techniques which are essential for locating appropriate business applications. WSCE enhances the process of discovering Web services by providing advanced search capabilities for locating acceptable business applications across one or more UDDI registries and any other searchable repositories.

2.2. Service Discovery

Service discovery, *i.e*., the automatic location of services that fulfill a user goal is a popular research topic. Service discovery is an essential requirement for any distributed, open, dynamic environment. An important step for fully open and flexible web service application would be the mechanization of service discovery. The Web Service Modeling Ontology (WSMO) provides the conceptual framework for semantically describing web services and their specific properties. The Web Service Modeling Language (WSML) is a formal language for annotating web services with semantic information, which is based on the WSMO conceptual framework.

A brief introduction is given on service discovery here. This chapter introduces the concepts of services and service discovery. Furthermore, it presents with the introduction of service and background of service discovery and basic definitions of service discovery.

2.3. Service

Services are ubiquitous. In everyday life, people use services in their environment. These services can be for instance physical services (*e.g*. the bakery on the corner, the usage of the microwave at home), social services and electronic services (E-services). Well-known examples of this last type of services are web search engines, route planners and electronic tax statements. Our research focuses on E-services. Service is nothing but work done for others. The database is a service that stores and retrieves data for client service is a software entity that performs an action on behalf of another entity.

A Service is a software entity provided by a Service Provider. It performs an action (based on inputs) on behalf of a Service Requestor and provides a result (output). An example of a service is the use of a web search engine: when a user (service requestor) types a query (input) into the search engine (service provider), the search engine tries to find relevant web sites (action) and returns the findings to the user (output). The operational lifecycle of a service consists of three successive phases: Advertisement, Discovery and Delivery. This is a generalization of the model proposed in [SUL02, KLE03]. In the advertisement phase, the service provider creates a service description based on the service properties. This description is used to advertise (*i.e*. enable the service for use) the service. In the discovery phase, the requestor tries to find (*i.e*. manual or automatic discovery) a service that satisfies his need. When a service is found, it is provided in the delivery phase.

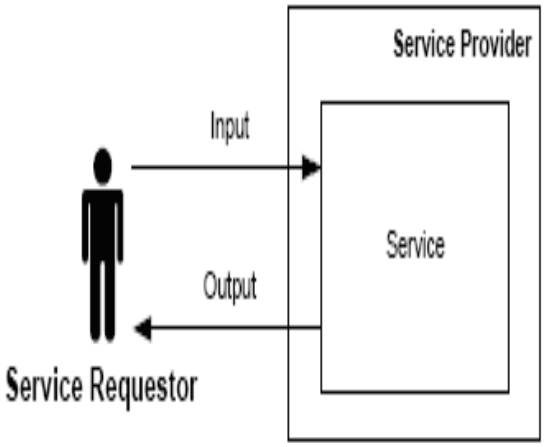

Figure 2.1. Service

Before the delivery phase, the service requestor and service provider are unassociated with each other (maybe even not aware of each other's existence). The service provider creates a service and advertises it to the world. The requestor tries to find some service it desires. With the service discovery the service requestor and some service provider are associated (depicted by bold lines in figure) with each other (*i.e*. they become aware of each other's existence) and both can participate in the delivery phase.

2.4. Background on Service Discovery

To use a service, it first has to be found. This is often difficult for a service requestor. Services are distributed and heterogeneous. Service discovery can be defined as the act of finding services. The link between the service provider and the service requestor (as presented in Figure 4) is not that obvious. Without a priori knowledge (*e.g*., location of the service, IP address) it is difficult for the service requestor to use the service it needs.

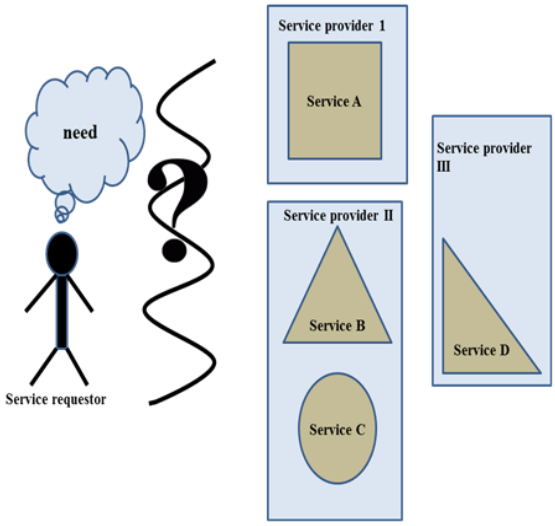

Figure 2.2. Service Discovery

2.5. Classification of Service Discovery

Goal discovery

Users may describe their desires in a very individual and specific way that makes immediate mapping with service descriptions very complicated. Therefore, each service discovery attempt requires a process where user expectations are mapped on more generic goal descriptions. Notice that this can be hidden by the fact that a discovery engine allows the user only to select from predefined goals. However, then it is simply the user who has to provide this mapping *i.e*. who has to translate his specific requirements and expectations into more generic goal descriptions.

Service discovery

Service discovery is based on the usage of web services for discovering actual services. Web service technology provides automated interfaces to the information provided by software artifacts that is needed to find, select, and eventually buy a real-world service or simply find the piece of information somebody is looking for.

Web service discovery

Web service discovery is based on matching abstracted goal descriptions with semantic annotations of web services. This discovery process can only happen on an ontological level, *i.e.*, it can only rely on conceptual and reusable.

Location aware service discovery

The capability of automatically finding the services, which is available in the nearby environment, is said to be location based service discovery. This discovery process can only happen using location sensing system.

P2p service discovery in web service

The key aspect of P2P based Web Services discovery is the query propagation in the peer networks, because that messages routing the topology of the network has significant impact on the overall performance of the service discovery process.

Various providers publish Service advertisements with embedded QoS information in P2P-based registries and users can query for services with certain functionalities and required QoS levels, using any registry peer as their access point. The P2P-based registries then take care of routing the request to the peer(s) that can answer it the results will be returned to the user and this user may invoke one of the found services.

Context aware service discovery

CA service discovery can be defined as: the act of finding services thereby using context. To use a CA service not solely an association between the service requestor and service provider (two parties) need to be created (traditional service discovery) but an association between the service requestor, service provider and context provider (three parties) needs be created.

Ontology based service discovery

Ontology's are a formal and explicit specification of a shared conceptualization that can be used for sharing and reasoning on knowledge. This approach is an envisioned approach to overcome issues with non-common vocabularies] important for service discovery like discussed in the motivation.

3. Problem Analysis

Service Discovery is done based on the following way.

- Service Discovery based on Keyword Match.
- Service Discovery based on Simple Semantic Description Services.
- Service Discovery based on Rich Semantic Description Service.

Web Services lift the Web to a new level by integrating computational aspects. Web Services are accessed by computer programs but these still need help by humans. Web-services should be described in a formal, semantic way, so programs can find, compose and invoke them automatically. Existing system discover the service based on QOS. It

uses the Agent to discover the service, but normalization & classification are not done by Agent in here. Existing system are using the selection algorithm, which works in association with QOS value, based on consumer requirements.

3.1. Issues in the System

Existing system discover the service based on QoS, It uses the Agent to discover the services, Existing systems are using selection algorithm which works in association with QoS value based on consumer requirements. The main issues here are that CLASSIFICATION, NORMALIZATION, & MATCHING RANKING are not done.

There are major problems in using QoS for web service discovery. First is the specification and storage of QoS information available. Mostly web service discovery process falls in two logical steps-

Web service matching, meets specific functionality required by user from existing service.

Web services selection, that is choosing a service with the best quality among those matching services.

3.2. Solution

Since are issue in the system is that normalization classification and ranking are not done. So we give a solution as a resultant in this method is to modify the parameters. Parameters like reliability, complaint best practice memory conception and classification.

Solution gives the best system reliable with the services and memory conception, *etc.* Gives a solution over a maximum number of services, like all kind of banking service, and searching. Match services like both functional as well as QoS, if QoS is not available in database in that case QoS will be randomly select the ranking; this is what the solution we give

4. Proposed Scheme

In our proposed system we proposed the Normalized, Classification, & Ranking of the QoS parameter based on minimization maximization criteria.

Normalization- We normalized according to the service ranking that is low and high value of QoS. We select the maximum high value and minimum of low value for require criteria. In which security will be provided according to their value.

Classification- We classified web service QoS based on Performance, Cost, & security. Mostly consumer prefers for high performance of Services.

Ranking- Ranking is an important framework which first classifies candidate's web services to different QoS levels according to QoS parameter requirement preference.

We rank the services according to their QoS levels and their services matching.

5. Implementation

5.1. System Architecture

They support for web services. It includes QoS description using Model during the service publication, and performs powerful QoS aware invocations. In addition it verifies, certifies, confirms, and monitors QoS properties dynamically via an agent-based web service. The architecture involves four main participating roles such as Web Services Agent (WSA), web services provider, web services consumer, in addition to a QoS enabled UDDI registry.

The WSA has four components: Service Publisher, Authenticator and Certifier, Discovery Agent and Web Services (WSS). The WSA assists the client for selecting web services based on a set of QoS properties. The agent is a web service performing a

multiple related object having partial as a group of QoS functionalities. It is the entity that performs the verification and certification tasks. It is also involved in other operation, such as registering and selecting services with QoS functions. The service publisher self contained part of a large entity facilitates the registration, updating and deletion of web services from the services provider. The services provider publishes its service ability to perform a task to the UDDI registry through the service or group of services there exist a service publisher that handles all communication with registries, binding, negotiations, requests and responses for that services.

The service consumer can search the UDDI registry for a specific service through the discovery agent. The main functionality of the discovery agent is to select the most suitable web service which satisfies requester's QoS constraints and preferences, along with service functionality. The service consumer can verify the advertised QoS through the discovery agent before binding the web service. The verifier and certifier component is implemented within the WSA and is responsible for certifying web services and their provided QoS.

The QoS property of quality obtained from the service providers are verified and before registering them into the UDDI registry. QoS verification is the process of validating the errors of information described in the service interface.

The result of verification will be issued when the verification succeed. A certificate is sent to the web service provider and a copy is stored in the WSS for future use.

A typical usage scenario is described here by considering an example in which a consumer uses a web service in its application. Initially Web Service Agent publishes its interface to the UDDI registry. Web Service provider finds the agent interface in UDDI registry.

The service provider registers the web service with the service publisher which is available in the WSA and providers functional and non-functional information about the proposal that has been a service.

The verifier and certifier component of the WSA verifies the QoS information and issues a certificate. A copy of the QoS certificate is stored in WSS and a copy is sent to the service provider.

The service publisher then one who publishes the web service in the UDDI registry along with the QoS certificate. The consumer application requests service discovery which provides functional and QoS requirements to discovery agent.

The discovery agent finds the service in the UDDI registry agreeing to the required service functionality and QoS requirements of the application. Discovery agent can communicate with WSS to verify the provided QoS certificate from the UDDI against with the stored certificate in WSS.

The discovery agent then reports the discovered service information back to the application. The web service consumer then binds the web service from the service provider.

5.2. Implementation of Web Service Discovery with Qos

We suggest the agent- based architecture for dynamic Web service discovery which facilitates the requester to specify his/her QOS requirements along with functional obligatory. We propose a novel approach for designing and developing an agent-based architecture and its QOS-based design, ranking and selection algorithm for evaluating web services.

5.3. Evaluation

In this phase the Comparison of traditional discovery system with the proposed web service discovery model is done. The evaluation of the services is done by specifying its unique WSDL address. Response time and accuracy of the system are compared to find

the efficiency of proposed model. From the Evaluation it is easy to conclude that, even though the web service discovery model is slow but provides accurate information.

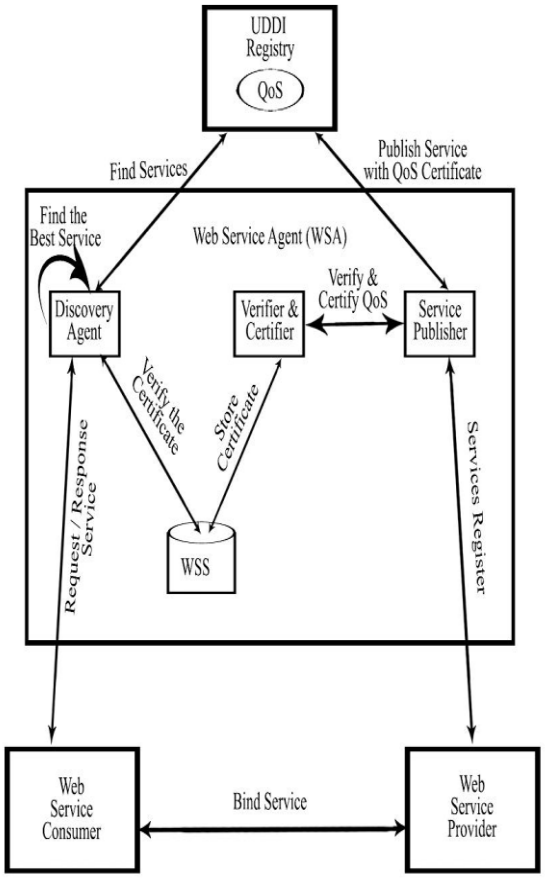

Figure 5.1. Architecture Diagram

6. Conclusion

We presented an agent-based architecture for web service discovery. The goal of the agent is to support web services discovery with QoS registration, verification, certification, and confirmation. The agent performs the publishing and selection of the web services. We describe the key features of the agent that are not supported by existing approaches dealing with QoS for web services. We propose an agent approach for dynamic web service discovery which hides the system complexity from the clients. An amount of web service is needed to test the performance of the system. This will enable a more flexible and trustable architecture. Result of this work will be reported in future paper.

7. Future Enhancement

Optimal agent is ready to provide all the functional properties of the web service then all the web service technique all the non functionalities. The goal of agent is to support web service fundamental with QoS. And its vision is a brokerage system that enables service to publish to a searchable repository and lather retrieved by potential user. This

includes agent based architecture for dynamic web service discovery which facilitates the requestor to specify his/her QoS requirement along with fundamental requirement.

An optimal agent based (OAB) architecture for dynamic web service discovery which facilitate the requester to specify his/her QoS requirement along with fundamental requirement. A novel approach for designing and developing an agent based architecture and each QoS based matching, ranking and selection algorithm evaluating web service. The paper presents an optimal approach for finding the most suitable web service according to the consumer requirement. We suggest theoretical architecture will be based on the implementation of QoS property.

Acknowledgment

We are thankful to the CARD SYSTEM of our college.

References

[1] J. Lu, T. W. Ling, Z. Bao and C. Wang, "Title-Extended XML Tree Pattern Matching- Theories and Algorithms", Publisher-IEEE Transactions on knowledge and Data Engineering, **(2011)**.
[2] J. Cheng, J. X. Yu and P. S. Yu, "Title-Graph Pattern Matching- A Join/Semijoin Approach", Publisher-IEEE Transactions on knowledge and Data Engineering, **(2011)**.
[3] W. Ouyang, F. Tombari, S. Mattoccia, L. D. Stefano and W. K. Cham, "Title-Performance Evaluation of Full Search Equivalent Pattern Matching Algorithms", Publisher- IEEE Transactions on Pattern Analysis and Machine Intelligence, **(2012)**.
[4] C. L. P. Chen, "Title- Automatic Assembly Sequences Generation by Pattern Matching", Publisher-IEEE Transactions on Systems, Man and Cybermatics, **(1991)**.
[5] A. Dembo and I. Kontoyiannis, "Title- Source Coding, Large Deviations, and Approximate Pattern Matching", Publisher-IEEE Transactions on Information Theory, **(2002)**.
[6] I. Kontoyiannis, "Title- Pattern Matching and Lossy Data Compression on Random Fields", Publisher-IEEE Transactions on Information Theory, **(2003)**.
[7] T. Łuczak and W. Szpankowski, "Title- A Suboptimal Lossy Date Compression Based on Approximate Pattern Matching", Publisher-IEEE Transactions on Information Theory, **(1997)**.
[8] N. Bruno, D. Srivastava and N. Koudas, "Title-Holistic Twig Joins-Optimal XML Pattern Matching", Publisher- IEEE Transactions on Information Theory, **(2002)**.
[9] C. Y. Chan, W. Fan and Y. Zeng, "Title- Taming Xpath Queries by Minimizing Wildcard Steps", Publisher- IEEE Transactions on Pattern Analysis and Machine Intelligence, **(2004)**.
[10] S. Chen, H. G. Li, J. Tatemura, W. P. Hsiung, D. Agrawal and K. S. Candan, "Title- Twig2stack-Bottom-Up Processing of Generalized-Tree-Pattern Queries over XML Document", Publisher- IEEE Transactions on Pattern Analysis and Machine Intelligence, **(2006)**.
[11] T. Chen, J. Lu and T. W. Ling, "Title- On Boosting Holism in XML Twig Pattern Matching Using Structural Indexing Techniques", Publisher- IEEE Transactions on Pattern Analysis and Machine Intelligence, **(2005)**.
[12] S. Chien, Z. Vagena, D. Zhang, V. J. Tsotras and C. Zaniolo, "Title- Efficient Structural Joins on Indexed XML Documents", Publisher- IEEE Transactions on Pattern Analysis and Machine Intelligence, **(2002)**.
[13] E. W. Dijkstra, "s``A note on two problems in connection with graphs", Number. Math, vol. 1, no. 1, **(1959)**, pp. 269-271.
[14] Z. Huang, J. Xu and Y. Yu, "Three-stage algorithm for dynamic service composition", Computer Integration Manufacturing System, vol. 18, no. 8, **(2012)**, pp. 1711-1718.

Authors

Suganya.D, pursuing M. Tech (computer science and engineering) in Rajiv Gandhi College of Engineering and Technology. Her area of specialization is web service.

Revathy.A, pursuing M. Tech (computer science and engineering) in Rajiv Gandhi College of Engineering and Technology. Her area of specialization is web service.

Suresh Kumar RG, Pursued MCA at Annamalai University in 2001. Pursued M. Tech (Information Technology) from AAD University in 2001. Currently pursuing Ph.D, in Computer Science & Engineering at Vels University. He is a Sun Certified Professional and His main research areas include Client Server computing, Cloud computing and Utility Computing.

N.Moganarangan pursued M. Tech at Dr. Paul's Engineering College. And he also pursued Ph.D, at MS University Thirunelveli. He is a Sun Certified Professional and His main research area is web service.

Madhavan.D, pursuing M. Tech (computer science and engineering) in Rajiv Gandhi College of Engineering and Technology. Her area of specialization is web service.