INDEXING TRADITIONAL UDDI FOR EFFICIENT DISCOVERY OF WEB SERVICES

K. Tamilarasi

Research Scholar, Department of Computer Science and Engineering,
Sathyabama University, Chennai.

Dr. M. Ramakrishnan,

Professor and Chairperson, School of Information Technology,
Madurai Kamaraj University, Madurai

Abstract

Web service is a major trend in the industry for loosely coupled service-oriented architecture and interoperable solutions across heterogeneous platforms and systems. It receives profound attention and adoption by the industry as well as standard bodies. The traditional web service discovery is based on one-way request/response interaction pattern that a client (service consumer) makes a service request to the server (service provider) based on the services specified in the server's WSDL. At the heart of service-oriented computing is a Web service registry that connects and mediates service providers with clients. The Universal Description, Discovery and Integration (UDDI) offers a standard way for businesses to build a registry, discover each other and describe how to interact over the Internet [9].UDDI is however limited in scope that searching if services is possible only by means of keyword which is not enough to justify the capabilities provided by the web services of different service providers. This paper proposes an indexing mechanism to be introduced in the traditional UDDI for effective discovery of web services which focuses on two QoS parameters namely response time and relevancy. Evaluation of the proposed framework is done by implementing the algorithm and performance is compared with the traditional UDDI discovery.

Keywords: UDDI, Discovery, Web service, Indexing, Search Engine

1. Introduction

Web services, are considered as self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. A Web service is an interface that describes a collection of operations that are network accessible through standardized XML messaging [10]. A major focus of web services is to make functional building blocks accessible over standard internet protocols that are independent from platforms and programming languages The Web Services architecture is based upon the interactions between three roles: service provider, service registry and service requestor. The interactions involve publish, find and bind operations. Together, these roles and operations act upon the Web Services artifacts: the Web service software module and its description [6].

The next step to consider is how to publicize this Web Service so that interested clients can easily discover and consume it into their applications. A discovery mechanism that fulfills this requirement exists today: UDDI, an industry-wide initiative to support Web Service discovery across technologies and platforms [3]. UDDI provides a worldwide registry of web services for advertisement, discovery, and integration purposes. Business analysts and technologists use UDDI to discover available web services by searching for names, identifiers, categories, or the specifications implemented by the web service. UDDI provides a structure for representing businesses, business relationships, web services, specification metadata, and web service access points [7]. UDDI is a standard designed to provide a searchable directory of businesses and their Web Services. Thus, it represents the service broker that enables service requesters to find a suitable service provider.

Most importantly; UDDI contains information about the technical interfaces of a business's services. Through a set of SOAP-based XML API calls, one can interact with UDDI at both design time and run time to discover technical data, such that those services can be invoked and used. In this way, UDDI serves as infrastructure for a software landscape based on Web Services. Taking a look at what data is stored in UDDI and how it is structured is the next step in understanding the UDDI initiative. UDDI is relatively lightweight; it is designed as a registry, not a repository. The distinction is subtle, but crucial. A registry redirects a user to resources, whereas a repository is an actual information store. The client queries a UDDI registry for the service either by name, category, identifier, or specification supported. Once located, the client obtains information about the location of a WSDL document from the UDDI registry [11]. The WSDL document contains

information about how to contact the web service and the format of request messages in XML schema. The client creates a SOAP message in accordance with the XML schema found in the WSDL and sends a request to the host (where the service is). To get access to the UDDI services, the UDDI directory exposes a set of APIs in the form of a SOAP-based Web Service. The core component is the UDDI business registration, an XML file used to describe a business entity and its Web Services.

The UDDI XML schema defines four core types of information needed by developers in order to bind to Web Services. They are business Entity (i.e., service provider), business Service (i.e., service), binding Template (i.e., access information) and tModel (i.e., service type definition).Business Entity describes information about business (e.g., names, description, services offered and contact data etc.). A business Entity contains a collection of business Service elements, one for each Web service that the business wishes to publish. Each business Service provides more details on the service offered by the service provider. A business Service contains a collection of binding Template elements. A bindingTemplate describes the access information (e.g., the endpoint address) and describes how the business Service uses various technical specifications. In other words, a bindingTemplate provides the details of how and where the service is accessed. A technical specification is modeled as a tModel. tModel provides the ability to describe compliance with a specification, a concept, or a shared design. In summary, UDDI provides two main functionalities: (1) A mechanism for service providers to register themselves and their services with the UDDI registry, (2) A mechanism for service subscribers to search for the available services from the UDDI registry.

UDDI merits are it services are ranked and filtered quickly. Its demerits are that the only possible way to discover web services is keyword based discovery and it contains only metadata and only single search criteria can be specified [9].

2. Related Work

Netra Patil et. al [8] conducts an extensive study on web discovery approaches based on centralized approaches and illustrates the importance of UDDI. Reza Rostami et. al [11] minimizes the response time by adapting web content based on user connection speed to ensure response time. T. Rajendran et. al [10] have made analysis on the study of QoS-aware web services discovery and focused on the QoS constraints to be followed during discovery of web services.

Eyhab Al-Masri et. al [5] presented many models on web service discovery and concluded that time effective and extremely useful means are necessary for finding services of interest and did crawling multiple UDDI registries. Antonio Brogi et. al [2] have presented a behavior-aware discovery of web service compositions and emphasized the importance of UDDI as the standard for web service discovery.

Wenli Dong et. al [16] implemented a QoS driven service discovery method in extended UDDI based on QoS and Ontology driven proposes solutions on decrease misunderstanding, increased discovery accuracy based on fuzzy correlation matching algorithm and a policy based discovery process based on Semantic Web.

Song et. al[12] discovered web service using general purpose search engines by inserting a WSDL specification in a Web page that offers semantic explanation with best results has been made evident by the experimental results of both search engines.

3. Web Service Discovery

Discovery Models: Service discovery approaches are classified into two broad categories syntactic based systems and semantic based model. Each classification model further has many system based on algorithm used, quality of Service, number of links, ontology and concepts usage.

Discovery Approaches: Web service discovery approaches can be classified as centralized and decentralized. Centralized approaches include search engines, web crawlers and UDDI as it uses centralized registry to store web service descriptions. Ad-UDDI, WSCE, METEOR-S, WSB are the other models proposed for centralized approach. Major web service providers like Microsoft, IBM and Amazon provide public UDDI [13]. Decentralized approaches employ ranking, semantic descriptions, Quality of Service, Cost of Service, match making algorithms, indexing to improve the efficiency of web discovery web services.

UDDI Extension Strategies: There are different strategies used for extending UDDI registries. First strategy is to use the same API and modifications can be done in the entry, query formats and its existing implementations. Second strategy is to modify the UDDI API with the new functionalities and the third strategy is to use middle tier to implement the extension and get connected to UDDI registry and service consumer.

Performance Metrics: The basic function of the UDDI is to find available web services matching requestors' demands. To evaluate the discovery process we use the following QoS metrics like available rate, success rate, average response time, and total traffic cost and the relevant QoS metrics are listed in Table 1.

Table 1. Relevant QoS metrics in Service Discovery.

Parameter	Description
Response Time	Time taken for a request to get a response
Availability	Number of successful invocations/total invocations
Throughput	Performance measured by total number of requests for a given period of time.
Successability	Number of response / number of request messages
Reliability	Percentage of successful invocation
Compliance	The extent to which a WSDL document follows WSDL specification
Latency	Delay time for a client request

4. Proposed Framework

There are different approaches for discovering web services namely UDDI, Service directories and portals and web search engines. Our contribution is to combine the features of UDDI with search engines for efficient discovery of web services [1].

The proposed framework integrates the traditional UDDI registry by introducing the indexing mechanism of search engines to minimize the response time of discovery of web services [14]. The proposed system is capable of providing the following advantages over the traditional UDDI registry, (1) most relevant web services to the user using index database minimizing response time, (2) consensus recommendation of web services to the user by mean of user preference database. It is based on two design choices and, they are (i) Indexer, (ii) Personalizer.

Proposed framework for efficient web service discovery mechanism is shown in fig 1 which includes the following components

UUI : UDDI User Interface is the component through which the service provider and service requestor publish and inquires the services.

UDDI Registry: It is the repository of web services. WSDL of all the registered web services are stored

Indexer: It is the additional work done to retrieve the web services in short response time. Inverted indexing is used to construct the index databases IB for business entity and IS for Business Service

Personalizer: This component takes care of the customization and user preferences to find out the most relevant services for the uses that uses the portal frequently. Preference Database DUP is constructed which consists of business log file and service log file.

The goal of developing Personalizer is to discover personalized businesses and their web services. The different processes involved in this module are

- Constructing of user preference database
- Grouping of businesses and business Services
- Service discovery using user preference database

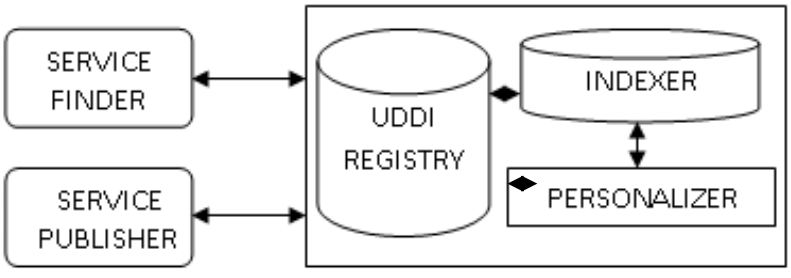


Fig. 1. Architecture of Efficient Web Service Discovery

The proposed web service discovery and selection algorithm is given below

Algorithm : ***Efficient Web Services Discovery***
Input: Request for Web service
Output: Desired Relevant Service
Services published in UDDI registries;
Indexing of web services – Index database
Personalization - Constructing of user preference database
Grouping of business and business services
User enters input request for web service;
for each input
Input goes to Search Engine;
Search Engine passes request to Personalizer;
Personalizer search from user preference database D_{UP}
If found
Results are returned
else
Search Engine passes request to Indexer; I_B and I_S
Indexer search from index database based on classification and indexing;
Search Engine adds references of matched services in user preference database D_{UP} through Personalizer for future reference;
Return results to user;
else
Return no service found;

5. Implementation

This section describes the prototype we have used for implementing an enhanced UDDI registry over traditional UDDI registry and consists of two different components, the enhanced UDDI registry and a UDDI client which acts both as service provider and requestor exploring the use of enhanced registry. The prototype is implemented using open source implementation of Java which is Apache JUDDIv3 and uses MYSQL database as repository for UDDI entries. For testing we have published around several businesses and services on registry. The implementation consists of service publishing, service registry, service inquiry, indexer and Personalizer. Around 300 sample services are deployed in Tomcat server to experiment and evaluate the proposed framework and the results have been depicted in Fig. 2 and Fig. 3.

Available services in :

JUDDI_Api_PortType • save_Clerk • get_publisherDetail • delete_ClientSubscriptionInfo • get_allPublisherDetail • adminDelete_tmodel • save_Node • invoke_SyncSubscription • delete_publisher • save_ClientSubscriptionInfo • save_publisher	Endpoint address: http://localhost:8090/juddiv3/services/juddi-api WSDL : {urn:juddi-apache-org:v3_service}JUDDIApiService Target namespace: urn:juddi-apache-org:v3_service
UDDI_CustodyTransfer_PortType • transfer_entities • discard_transferToken • get_transferToken	Endpoint address: http://localhost:8090/juddiv3/services/custody-transfer WSDL : {urn:uddi-org:v3_service}UDDICustodyTransferService Target namespace: urn:uddi-org:v3_service

UDDI_Inquiry_PortType • get_bindingDetail • find_service • get_serviceDetail • get_tModelDetail • find_binding • find_business • get_businessDetail • find_relatedBusinesses • find_tModel • get_operationalInfo	Endpoint address: http://localhost:8090/juddiv3/services/inquiry WSDL : {urn:uddi-org:v3_service}UDDIInquiryService Target namespace: urn:uddi-org:v3_service
UDDI_Publication_PortType • save_tModel • delete_binding • delete_business • add_publisherAssertions • save_service • set_publisherAssertions • delete_publisherAssertions • get_publisherAssertions • delete_service • save_binding • save_business • get_registeredInfo • get_assertionStatusReport • delete_tModel	Endpoint address: http://localhost:8090/juddiv3/services/publish WSDL : {urn:uddi-org:v3_service}UDDIPublicationService Target namespace: urn:uddi-org:v3_service
UDDI_Security_PortType • discard_authToken • get_authToken	Endpoint address: http://localhost:8090/juddiv3/services/security WSDL : {urn:uddi-org:v3_service}UDDISecurityService Target namespace: urn:uddi-org:v3_service
UDDI_SubscriptionListener_PortType • notify_subscriptionListener	Endpoint address: http://localhost:8090/juddiv3/services/subscription-listener WSDL : {urn:uddi-org:v3_service}UDDISubscriptionListenerService Target namespace: urn:uddi-org:v3_service
UDDI_Subscription_PortType • get_subscriptionResults • get_subscriptions • delete_subscription • save_subscription	Endpoint address: http://localhost:8090/juddiv3/services/subscription WSDL : {urn:uddi-org:v3_service}UDDISubscriptionService Target namespace: urn:uddi-org:v3_service

Fig. 2. Screenshot of the search engine-based UDDI after inputting the business query

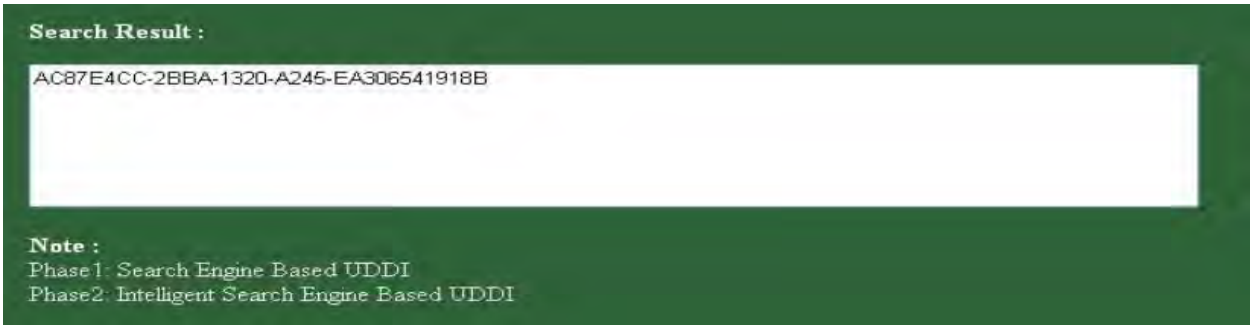


Fig 3. Screenshot of the intelligent search engine-based UDDI after inputting the business query

6. Evaluation and Analysis

Our efficient algorithm discovers the relevant web services in minimum response time and the user is not frustrated as the relevancy of discovered web services is compatible to the users. The performance is measured by means of relevancy and response time.

The proportion of services which meets the user requirements from the net discovered services is used for calculating relevancy factor. We have calculated the time taken to return the results by varying the no of services up to maximum of 25 services and the results have been given in Fig 4 and Fig 5. From the analysis, it has proved that the response time is greatly reduced by introducing the indexing in to the UDDI registry.

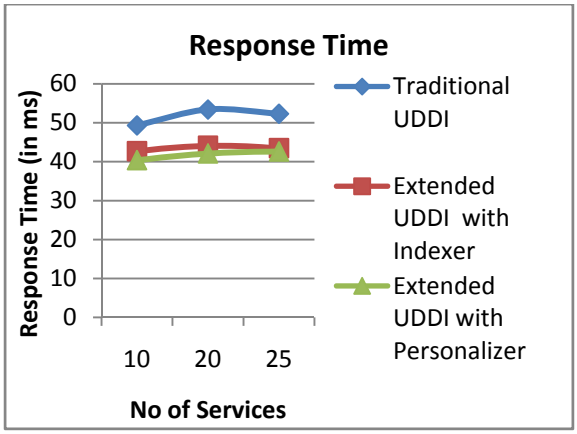
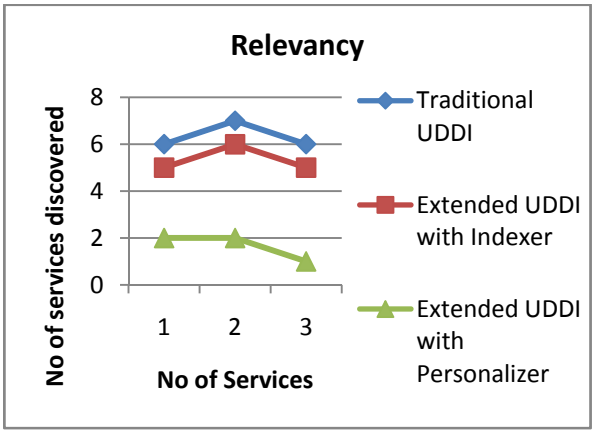


Fig. 4. Response time of Web services



Fig, 5. No of Relevant services returned

7. Conclusion and Future Work

In this paper, an efficient framework is presented for efficient discovery of web services. This framework can be easily extended for other QoS parameters. Proposed algorithm retrieves and minimizes the response time of discovery. Also only relevant web services are fetched to the requestor thus eliminating the irrelevant web services. In future, the extensions can be done to the framework by adding algorithms based on Intelligent and fuzzy techniques.

8. References

[1] Alexander Mintchev, (2008): "Interoperability among Service Registry Implementations: Is UDDI Standard Enough?", Proceedings of the IEEE International Conference on Web Services (ICWS '08), pp. 724 – 731, Beijing.

[2] Antonio Brogi, Sara Corfini, (2007): "Behavior-Aware Discovery of Web Service Compositions", International Journal of Web Services Research (IJWSR).

[3] Chen Zhou, Liang-tien Chia, Bu-sung Lee, (2004): "QoS-Aware and Federated Enhancement for UDDI" in International Journal of Web Services Research.

[4] Esmiralda Moradian, Anne Hakansson, (2006): "Possible attacks on XML Web Services", IJCSNS International Journal of Computer Science and Network Security, Vol.6 No.1B.

[5] Eyhab Al-Masri and Qusay H. Mahmoud, (2007): "Crawling multiple UDDI business registries", Proceedings of the 16th international conference on World Wide Web, pp. 1255 – 1256, Banff, Alberta, Canada.

[6] Juric. M.B. et.al. (2009): "WSDL and UDDI Extensions for Version Support in Web Services", Journal of Systems and Software, 82, pp. 1326–1343

[7] Justin Zobel, Alistair Moffat, (2006): "Inverted files for text search engines", ACM Computing Surveys Vol 38, No.2. Article 6.

[8] Netra Patil, Dr. Arpita Gopal, (2011): "Comparative study of mechanisms for web service discovery based on centralized approach focusing on UDDI", International Journal of Computer Applications (0975 – 8887).

[9] Preeti Marwahaa, Hema Banatib, Punam Bedic, (2013): "UDDI Extensions for Temporally Customized Web Services", Elsevier Publications.

[10] Rajendran.T, Balasubramie. P, (2009): "Analysis on the study of QoS-Aware Web Services Discovery", Journal of Computing.

[11] Reza Rostami, Thiam Kian Chiew, (2013): "Adapting Web Content Based On User Connection Speed To Ensure Response Time", Malaysian Journal of Computer Science. Vol. 26(4).
[12] Song, H. Doreen Cheng Messer, A. Kalasapur, S.,(2007): "Web Service Discovery Using General-Purpose Search Engines", In Proceedings of IEEE International Conference on Web Services, Salt Lake City, UT, pp.265 – 271.
[13] Tamilarasi, K. M. Ramakrishnan, (2012): "Design and Development of an Enhanced UDDI for Efficient Discovery of Web Services", Advances in Communication, Network, and Computing Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering Vol: 108, pp: 109-114.
[14] Tewari, V et.al. (2009): "An Improved Discovery Engine for Efficient and Intelligent Discovery of Web Service with Publication Facility", Proceedings of World Conference on Services – II, Bangalore.
[15] Urjita Thakar, Nirmal Dagdee, (2009): "An Approach to Discover Web Service Providers Based on Security Support", International Journal of Web Applications, Vol.1, No.2.
[16] Wenli Dong, (2007): "QoS Driven Service Discovery Method Based on Extended UDDI", In Proceedings of Third International Conference on Natural Computation, Haikou, Vol.5, pp.317 – 324.