Web Service with Criteria: Extending WSDL

Parimala N
School of Computer and Systems Sciences,
Jawaharlal Nehru University,
New Delhi, India
dr.parimala.n@gmail.com

Anu Saini
School of Computer and Systems Sciences,
Jawaharlal Nehru University,
New Delhi, India
anuanu16@gmail.com

***Abstract*. WSDL is used to describe the interface of a service, in XML format. The interface describes the functional properties as well as non functional properties. We are concerned with specifying 'criteria' as a non functional property of a web service. For this we have extend WSDL to X-WSDL. In order to add criteria information we extend the WSDL (Web Service Definition Language) schema by adding a new element 'criteriaservice' this is available in the new namespace. Using this 'criteriaservcie' element it is possible to specify the criteria along with a service in an X-WSDL document. The WSDL document is also extended by adding new attributes 'criteria name' and 'description' to service element. Using this extension it is possible to specify the criteria along with the service in X-WSDL document. The criteria are specified by the user when invoking a service. As a result, we are providing support to discover a more appropriate service according to his/her requirement.**

Keywords: Service Oriented Architecture (SOA), Web Service, UDDI, WSDL, X-UDDI, X-WSDL

I. INTRODUCTION

Web service is defined as a piece of business logic which is available on the internet and that can be accessible with the help of internet protocols [1, 2, 3, 4]. The three basic elements of Web Service are WSDL which is used for the description of the web service, Simple Object Access Protocol (SOAP) which is a transport protocol for exchange of information and lastly the Universal Description, Discovery and Integration (UDDI) which is a registry and is used to store the service.

Web service has three characteristics: functional, behavioral and non-functional [5]. Functional description tells us about what exactly the service can do. Behavioral description details how the web service works and how it can be integrated using Orchestration and Choreography. Nonfunctional descriptions are the constraints on the functional properties which are given by the user to discover the service. Non Functional Properties (NFPs) are quality of service (QoS), performance, scalability, reliability, availability, stability, cost, completeness etc [6].

The NFP that we want to associate with a web service is referred to as 'Criteria'. In order to bring out the meaning of criteria, consider an example where the user is searching for a hotel for booking a room. The choice of the hotel may not necessarily depend on the basic information like name of the hotel, phone number, etc. But it may depend on some facilities like internet connection, swimming pool etc. Such additional information is referred to as criteria. It is desired that this additional information be combined with the functional property and it should be possible to publish and invoke the web service with the criteria.

A number of efforts have been made to describe and discover service based on the non-functional properties. Invariably, the search for a web service based on NFPs is by using a broker/agent/mediator/reputation manager framework [7, 8, 9, 10]. In this approach the consumer sends a request (containing the functional and NFPs) to the Web Service broker/agent/mediator/ reputation manager. The Web Service broker/ agent/ mediator will respond with the address of the best fitting Web Service. Then the consumer can access the Web Service directly, using the provided address. For example, in [11] the non-functional property, QoS, is described using semantic framework and a reasoner is built to discover the Web services using the semantic framework. In UDDIe [12], the standard UDDI is extended to UDDIe to include a property bag which describes the properties of a service. The search based on properties is implemented within a broker framework.

The broker based approach has a number of limitations. Firstly, when users send large number of requests, the broker may become a bottleneck. Secondly, broker based approach provides the best fit option to the user, which is desirable in most of the cases but may not be desirable in all the cases. As an example, consider booking a room where the user may want more than one option so that he/she can make the final choice rather than allow the system to give the best option. Lastly, the broker architecture is built over and above the standard architecture of SOA and is not a part of it.

To overcome above mentioned limitations, we propose an approach which is different from the above broker/agent/mediator/reputation manager framework. In our proposal, there is a direct interaction between the UDDI registry and the user. As a result the user is able to get a list of services so that he/she can take the final decision. And we also propose that the specification of criteria and the functional aspect of the service be within the framework of the basic SOA architecture. In SOA architecture firstly, service provider describes the service using WSDL and saves it in the UDDI registry. Then the consumer sends a query to UDDI to determine the required service. The address of the service provider from the WSDL description is passed on to the service consumer according to his/her query. Then, the consumer uses this WSDL information to send a request to the provider. Finally, the service provider sends a reply to the consumer.

To incorporate the criteria in the web service within the above framework, we need to extend the standard UDDI and WSDL. In our earlier work [13], we have extended the UDDI to X-UDDI which is used to incorporate criteria in Decision Support Web Service (DSWS). X-UDDI contains a list of

978-1-4577-1539-6/11/$26.00 ©2011 IEEE

criteria which is stored in the extended orange page with its description. This orange page is associated with criteria bag.

In this paper, we deal with extending the WSDL to X-WSDL, to incorporate criteria as non functional property. For X-WSDL firstly we have to extend the standard WSDL schema by adding new keynames. Existing WSDL contains six elements named as definitions, type, message, porttype, binding and service. We extend two of these, namely, definition and service. By extending these elements in WSDL we include the additional criteria information into the existing WSDL. A new namespace is provided, so that user can easily use the extended features.

The layout of the paper is as follows. A survey of related work is dealt with in section II. In section III the proposed approach is explained. Section IV contains the Case Study. Section V is the concluding section.

II. RELATED WORK

In recent times, several researchers have extended the WSDL to specify properties which cannot be expressed using the standard WSDL. Many efforts have been made to extend WSDL. These range from specifying security requirements, QoS, performance, testing parameters to handling versioning.

Non Functional Properties (NFPs) are specified by using the extension element of WSDL in [14]. Here the author gives end to end support of NFP from description to measurement and updating of values. This paper uses only extension element to add NFPs.

WSDL and UDDI are extended to support the development time and runtime versioning of web service in [15]. This extension is used to maintain various versions of single as well as multiple service interfaces. They extend WSDL at service level as well as at operation level. Operation level versioning allows detailed control over types, messages, interface, bindings, and service endpoints. Service level versioning is used where WSDL changes is required to the interface, types, messages, and other elements at the same time. But operational level versioning is time consuming and it requires a careful planning.

WSDL is extended with the non functional properties to make the mapping between physical object and web service more accurate with the help of Model Driven Architecture in [16]. This architecture contains WSDL metamodel extension and transformation between UML models and XML syntax. Here a physical object is defined in the form of web service by extending the WSDL. But this extension is not applicable to complicated objects, like the objects which are having comprehensive concept with tremendous information. And this model is complicated if it is extended for every NFPs. This model is suitable for composed services instead of some single ones.

SOA (Service Oriented Architecture) and EDA (Event Driven Architecture) concepts are combined by using the extension of WSDL and BPEL in [17]. This paper provides support for business events and EDA (Event Driven Architecture) concepts in SOA. They extend the web service in such a way that the service is able to produce and receive events. For this they add some new constructs in WSDL.

Performance of the web service is predicted by using WSDL extension [18]. Here, WSDL is extended to predict the performance of a web service. They give the notation of P-WSDL as lightweight extension of a WSDL. The authors first introduce the WSDL metamodel which is derived from the WSDL XML schema and then transform it into performance enabled WSDL (P-WSDL).

WSDL is extended for the testing of web service in [19]. For regression and blackbox testing they extend the WSDL with Input-output Dependency, Invocation Sequences, Hierarchical Functional Description and Concurrent Sequence Specifications.

In [20], UDDI and WSDL are extended for the security of web services are extended the schema of WSDL and UDDI for publishing and discovering of the web service. The security parameters in the schema of WSDL and UDDI as optional parameters are added. But to provide a secure environment in which transactions can occur, the UDDI schema and WSDL/WSFL (Web Services Flow Language) definitions need to be extended to provide linkages to infrastructures relevant to trusted business transaction [Extending WSDL to Facilitate Web Services Testing].

The standard WSDL metamodel is transformed in to QoS-enabled WSDL metamodel (Q-WSDL) [21]. An additional transformation is performed to get the Q-WSDL XMLSchema. By these transformation users get the extended WSDL i.e. Q-WSDL with several quality attributes.

WSDL is extended to WSDL-S to incorporate the QoS in [22]. Here two elements are added to define the qualities that are qualityIndependent and qualitySpecific. The semantic concepts are defined in ontologies. The upper-ontology is built as the basis of quality ontologies. In upper ontology independent quality attributes are defined. Even though WSDL-S has major benefits, there are some drawbacks. Ontologies are not flexible, difficult and complex to define [23, 24]. Making a consistent and appropriate ontology and its further maintenance is another big problem in the usage of semantic web service [24]. Further, it is difficult for the user to understand the semantics as they are complex and difficult to use.

Even though we also extend WSDL we are different from all the above proposed extensions to WSDL. In [21], the extension of WSDL to Q-WSDL is a multistep model driven process using the metamodel whereas our approach is a single step process. In [15] the extension to WSDL is for version management and is not applicable for our system as we work with a single version. We are different from semantic web service as we do not associate any meaning to a service but enhance the service description with non functional properties. In our proposal we extend WSDL to be able to specify criteria along with a service description. X-WSDL will support a service search, which takes the criteria from the user and finds services according to these criteria.

III. Proposed Approach

In SOA architecture, the functional properties are published and invoked. As discussed above, the specification of criteria is added within the architecture of SOA. That is, the pattern of register, search and invoke service is maintained. However, WSDL and UDDI are modified to accommodate the specification of criteria. Following steps are used to add criteria in the web service:

A). **Specify criteria by using X-WSDL**: The provider of the service specifies the criteria in the description of the web service using X-WSDL.

B). **Publish the criteria associated service:** Publish the X-WSDL description in the X-UDDI registry.

C). **Invoke the criteria associated service:** Client searches in the X-UDDI to find the appropriate web service according to the desired criteria.

The above steps are pictorially depicted in Figure 1.

A. Specify criteria by using X-WSDL

The criteria have to be specified along with the service in the WSDL document. In order to be able to do so the elements required for the specification of the criteria must bedefined in the WSDL schema. We therefore first extend the WSDL schema and then show the extension to the WSDL document.

I) Extending WSDL Schema

In this section, we explain the specification of criteria within the X-WSDL schema. To extend the schema new key element is added in the schema. This definition is available in a new namespace http://localhost:8080/X-UDDI/wsdl1.

Under the **definition** element we introduce a new key criteriaservice as shown in figure 2.

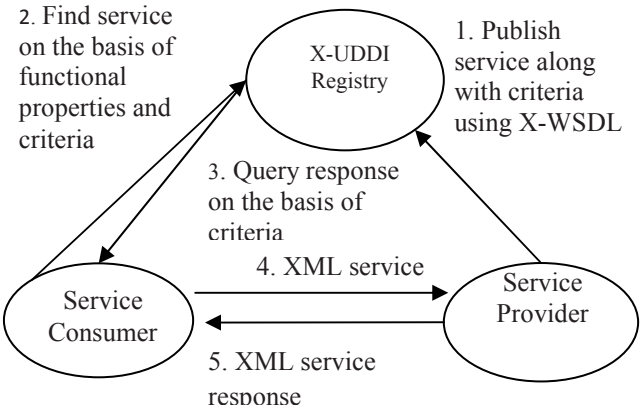

Figure 1. Modified SOA Architecture with X-WSDL

```
<xs:element name="definitions">
        <xs:key name="criteriaservice">
        <xs:selector xpath="cr:service"/>
        <xs:field xpath="@name"/> </xs:key
</xs:element>
```

Figure 2. Schema definition for X-WSDL

The criteriaService definition as a complex data type is given in figure 3:

```
<xs:complexType name="criteriaService">
   <xs:complexContent>
   <xs:extension base="cr:cExtensibleDocumented">
  <xs:sequence>

<xs:elementname="criteriaport"type="cPort"minOccurs="0"
      maxOccurs="unbounded"/>
 </xs:sequence>
       <xs:attribute name="name" type="xs:NCName"
          use="required"/>
       <xs:attribute name="criteria" type="xs:string"
          use="optional"/>
        <xs:attribute name="description" type="xs:string"
          use="optional"/>
 </xs:extension>
</xs:complexContent>
</xs:complexType>
   <xs:complexType name="cPort">
<xs:complexContent>
<xs:extension base="cr:cExtensibleDocumented">
   <xs:attribute name="name" type="xs:NCName"
   use="required"/>
   <xs:attribute name="binding" type="xs:QName"
          use="required"/>
   </xs:extension>
</xs:complexContent>
</xs:complexType>
```

Figure 3. Service specification in X-WSDL schema

As can be seen below criteriaService contains a sequence of criteriaport elements. It also contains three attributes-service name, criteria and description. These attributes are used to specify the criteria and its related information to invoke the service.

II) X-WSDL Document

Having extended X-WSDL schema it is now possible to extend the elements of WSDL documents. Before doing so, we briefly describe the WSDL structure which is relevant to the extension.

WSDL forms the basis for Web Services. According to W3C, WSDL which is in XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information [25]. All the information about service capabilities and invocation mechanisms is described by using the WSDL. Service provider and service consumer can communicate with the help of WSDL. WSDL document uses the following elements in the definition of network services [26], as shown in figure 4a.

Definition- it is a root element of the WSDL which contains the name of web service and namespace
Types– a container for data type definitions using some type system (such as XSD).
Message – an abstract, typed definition of the data being communicated.
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 this paper, we are extending the WSDL at the service level. Doing so is adequate as our search is based on criteria associated service. Therefore, we extend the **service** element of the standard WSDL to support the additional feature of criteria and description as shown in figure 5b. In this paper we are extending WSDL 1.1 version.

Service is a collection of ports where port is the endpoint which is the collection of binding and service access address. We extend service element to include the criteria information along with the ports and bindings. The structure of X-WSDL document is shown in figure 5, with the extension of service element.

The **criteria name** attribute provides a unique name to every criterion defined within the service element. Service element may be containing more than one criteria name. Criteria name is the name of the non-functional property. This helps the user to find the more appropriate service by specifying the criteria.

The **description** attribute provide the detailed description of the criteria attribute. This is used to specify the complete requirement of the user in the documented form.

Definition element in WSDL is the root element which specifies the name of the web service and the multiple namespaces used throughout the document. The new namespace http://localhost:8080/X-UDDI/wsdl1 which contains the definition of critetiaservice has to be specified along with other namespaces.

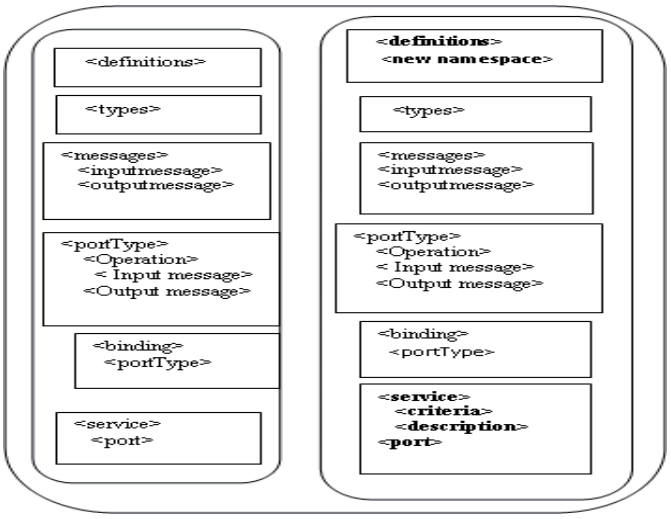

Figure 4a: Elements of WSDL Figure 4b: Elements of X-WSDL

```
<definition name=" "
   targetNamespace="http://localhost:8080/X-UDDI/wsdl1"
     xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
     xmlns:cr=" http://localhost:8080/X-UDDI/wsdl1"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <message name=" ">
          <part name=" " type=" "/>
      </message>
      <portType name=" ">
         <operation name=" ">
            <input message=" "/>
            <output message=" "/>
         </operation>
      </portType>
     <binding name=" " type=" ">
   <operation name=" ">
        <soap:operation soapAction=" "/>
        <input>
          <soap:body  encodingStyle=" "/>
        </input>
        <output>
          <soap:body encodingStyle=" "/>
        </output>
    </operation>
   </binding>
<cr:service name=" ">
       <criteria name=" "
       <description=" "
<port binding=" " name=" ">
       <soap:address  location="">
</port>
</service>
   </definition>
```

Figure 5. WSDL document with extended elements

B. *Publish the criteria associated service*

Web service interface is defined by using WSDL and this definition of the service is published in the UDDI, from where the customer can find the desired service. Similarly in our case, Service is defined by using X-WSDL and it should be published in X-UDDI [9].In our earlier work [13], UDDI is extended to X-UDDI by adding one more page (orange page). The page is used to save the criteria which are associated with the service.

This page is flexible and can be easily modified and updated by the provider. We also introduced one more bag i.e. criteria bag. Criteria bag is used to define the attributes of the criteria. When the service is searched for, the user has the option of specifying whether there can be a partial match or whether there must be a full match between the attributes specified by the user and those stored with the service. Towards this, we have introduced the attribute criteriaMatch with two values, namely fullCriteriaMatch and partialCriteriaMatch.

C. *Find the service based on criteria*

Once the service is published in the UDDI, then the user will start searching for the desired service in the UDDI. For invoking the service from the UDDI we need inquiry APIs. The Inquiry API is used by the clients to access the web service. Inquiry APIs consists of find_business, find_service, find_binding, find_tModel, get_businessDetail, get_serviceDetail, get_bindingDetail, and get_tModelDetail. To find the service based on the criteria we have introduced new find_dservice API in our previous work [9]. The inquiry API, find_dservice, allows the user to locate and obtain the service which matches the criteria list.

IV. Case Study

Consider the room booking example, where the traveller wants to book a room in a hotel on the basis of two criteria that are balcony and internet connection in the room.

In the figure 4, the user's two criteria are incorporated in the WSDL document using the newly added elements, criteria name and description, of WSDL schema. Here in this example, we only give the detailed description of the extended elements.

To start with the namespace http://localhost:8080/EUDDI/wsdl of the X-WSDL where the attributes of criteriaservice (cr:service) element are defined is included. The criteria, which are 'balcony' and 'internet', are the values of the criteria name attribute. This is followed by the description attribute as shown in figure 6.

Next step is to publish the X-WSDL of the service in X-UDDI. The definition of the service which is associated with the criteria is published in X-UDDI. Suppose Taj-Hotel_Service is providing two facilities, balcony and internet, which are to be associated as criteria with the service. The code for publishing Taj-Hotel_Service with criteria list is given in figure 7.

```
<wsdl:definitions
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
   xmlns:cr=" http://localhost:8080/EUDDI/wsdl1"
 targetNamespace="http://localhost:8080/EUDDI/wsdl1"
  <cr:service name="RoomBookService">
     <cr:criteria name= "balcony"/>
      <cr:description name =" room with balcony"
      </criteriadescription">
      <cr:criteria name= "internet"  element=
           "cr:bookaroomcriteria2"/>
      <cr:description name = " internet should be
       available in room" element= "cr:Bookroom
      </criteriadescription">
   <cr:port name="RoomBook" binding="  "> </cr:port>
</cr:service>
</definition>
```

Figure 6. Example of Definition of X-WSDL

```
<save_dService generic="2.0" xmlns="  ">
<businessService businessKey="*****" serviceKey="">
      <name>Taj_Hotel_Service</name>
    <criteria Bag>
< criteria >
      < criteria Name>balcony</ criteria Name>
      < criteriaDescription> "Room is available with
        spacious balcony service " </ criteriaDescription>
</criteria >
< criteria >
      <criteria Name>Internet </ criteria Name>
      < criteriaDescription> "Internet in room"</
       criteriaDescription>
</ criteria >
   </ criteriaBag>
</BusinessService>
</save_dService>
```

Figure 7. Publishing the service in X-UDDI

In figure 8, find_dservice API returns a list of services along with the service key, business key, criteria key and list of criteria that matched either partial or full depending upon the user's choice.

```
<find_dservice generic='1.0' xmlns='urn:uddi-org:api'
       businessKey=" ">
     <name>UDDI Web Services</name>
   <criteriaMatch>Full_criteria_match</criteriaMatch>
        <criterianame name= "balcony"/>
        <criteriadescription name =" spacious balcony"/>
        <cr:criterianame name= "internet"/>
       <criteriadescription name ="internet in the room"/>
</find  dservice>
```

Figure 8. Finding the service in X-UDDI

V. Conclusion

WSDL description is used by the web service providers to describe and publish services and it is used by the requester to find service. Service can be published as well as discovered on the basis of functional and non-functional properties. Service discovery and publishing is improved considerably by means of an effective WSDL extension with flexible and more suitable service discovery facility.

In this paper we have extended the WSDL to X-WSDL for publishing a service. We have extended WSDL to X-WSDL by extending the schema. To do so, new keynames in the schema have been added. After extended the schema, we also extend the WSDL document. For this we extend the service element of WSDL document. To add the criteria information with the service, two elements of WSDL namely criteriadefinition and critreiaservice have been added to the service element.

In this paper, we have explained our approach using version WSDL 1.1. However, the extension is equally applicable to version WSDL 2.0. These two versions are different while describing the interface. WSDL1.1 contains the <porttype> to describe the interface. Here, XML elements are used instead of input and output message. Interface uses operation name directly in the WSDL2.0 description. The element <port> in WSDL1.1 is replaced by the element <endpoints> in WSDL 2.0.

The two elements <binding> and <service> have not changed. They remain the same in both WSDL1.1 and WSDL2.0. In our case, we extend the <service> element. Therefore, we can use any version to incorporate the criteria information with the WSDL.

Our next step is to provide composable criteria based services. Even though WSDL1.1 is compatible with BPEL and WSDL2.0 is compatible with Choreography Description Language, it will be possible for us to explore composition using both, orchestration and choreography. The complete system will be evaluated with real life examples.

References

[1] Chappell, D., Jewell, T," Java Web Services", 1st edn. O'Reilly, Sebastopol (March 2002).

[2] Srivastava, B., Koehler, J.," Web Service Composition - Current Solutions and OpenProblems", ICAPS (2003).

[3] Alonso G., Casati F., Kuno H., and Machiraju V. "Web Services: Concepts, Architectures and Applications. Springer, 2004

[4] R. Jaya Prakash, R. Vimal Raja, " Evaluating Web Service Composition Methods With The Help Of A Business Application", IJECT Vol. 2(7), 2010, 2931-2935

[5] Toma, I., Foxvog, D, "Non-functional properties of web services" WSMO working draft (October 25, 2006).

[6] Yan-ping Chen, Zeng-zhi Li, Qin-xue Jin, and Chuang Wang, "Study on QoS Driven Web Services Composition. X. Zhou et al. (Eds.)" APWeb 2006, LNCS 3841, pp. 702 – 707, Springer 2006

[7] T.Rajendran, Dr.P.Balasubramanie, Resmi Cherian, "An Efficient WS-QoS Broker Based Architecture for Web Services Selection", International Journal of Computer Applications Volume 1 – No. 9, 2010

[8] M.Adel Serhani, Rachida Dssouli, Abdelhakim Hafid, Houari Sahraoui, "A QoS broker based architecture for efficient web services selection, IEEE (ICWS'05)

[9] R.Jeberson Retna Raj, T. Sasiprabha, "Web Service Selection Based on QoS Constraints", TISC, 2010 , vol., no., pp.156-162, 17-19 Dec. 2010

[10] D. A. DMello, V. S. Ananthanarayana, "Quality Driven Web Service Selection and Ranking", Fifth International Conference on Information Technology:New Generations.Volume 5, pp. 1175–1176., Apr. 2008

[11] Kyriakos Kritikos and Dimitris Plexousakis, " QoS-Based Web Service Description and Discovery", ERCIM NEWS 72, Special theme The Future Web, January 2008

[12] ShaikhAli, A., Rana, O.F, Al-Ali, R., Walker, D.W., " UDDIe: An Extended Registry for Web Services", SAINT Workshops. IEEE Computer Society, Washington (2003)

[13] N. Parimala, Anu Saini, " Decision Support Web Service",: ICDCIT 2011, LNCS 6536, pp. 221–231, © Springer-2011

[14] Agarwal, V.; Jalote, P, "Enabling end-to-end support for non-functional properties in web services", IEEE SOCA, pp.1-8, 14-15 Jan. 2009

[15] Matjaz B. Juric, ana sasa, Bostjan Brumen, Ivan Rozman, " WSDL and UDDI extensions for version support in web services", Journal of Systems and Software Volume 82, Issue 8, Pages 1326-1343, August 2009

[16] Changying Dai, Zhibin Wang, "A flexible extension of WSDL to describe non-functional attributes" EBISS, vol., no., pp.1-4, 22-23 May 2010

[17] Matjaz B. Juric, "WSDL and BPEL extensions for Event Driven Architecture" Information and Software Technology Volume 52, Issue 10, Pages 1023-1043, October 2010

[18] Andrea D'Ambrogio and Paolo Bocciarelli. " A Model-driven Approach to Describe and Predict the Performance of Composite Services", WOSP '07 ACM, 2007

[19] W. T. Tsai, Ray Paul, Yamin Wang, Chun Fa, and Dong Wang, "Extending WSDL to Facilitate Web Services Testing" 7th IEEE International Symposium HASE'02

[20] Carlisle Adams, Sharon Boeyen, "UDDI and WSDL Extensions for Web Services: A Security Framework", ACM Workshop on XML Security, Fairfax, VA, USA, Nov. 22, 2002

[21] Andrea D'Ambrogio, "A Model-driven WSDL Extension for Describing the QoS of Web Services", ICWS'06

[22] Keshan Zhu, Zhenhua Duan and Jianli Wang, "Quality of service in web services discovery", IEEE 2008

[23] Mossab A. Al Hunaity, "Towards an Efficient Quality Based Service Discovery Framework", IEEE Congress on Services, 2008

[24] Konstanty Haniewicz, Monika Kaczmarek, Dominik Zyskowski, "Semantic Web Services Applications- A Reality Check", WIRTSCHAFTSINFORMATIK , Gabler Verlag, 2008

[25] Erik Christensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, "Web Services Description Language (WSDL) 1.1", W3C Note 15 March 2001

[26] WSDL 2.0: A Pragmatic Analysis and an Interoperation Framework, 2008 SYS-CONMedia,