A Quality of Service (QoS) Constraint Language for Web Service Selection

Demian Antony D'Mello, and V. S. Ananthanarayana

Abstract--**Quality of Service (QoS) in Web services is a decisive factor for the success of this rising Web technology. Quality of Service (QoS) can be used to discriminate and rank functionally similar Web services. In this paper we propose a QoS model for Web service selection which categorizes QoS properties based on requester's selection point of view as business specific, performance specific, requester's response specific and service specific QoS properties. The paper defines the term QoS constraint and explores different types of requester's QoS constraints. The paper proposes tree representation and an XML structure for requester's various QoS constraints. We also propose the QoS based Web service selection model which finds the qualitatively best Web service based on requester's QoS constraints.**

Index Terms—**Information Services, Distributed Computing, Modeling, Quality of Service.**

I. Introduction

A Web service is an interface that describes a collection of operations that are network accessible through standardized XML messaging [1]. Web services can be advertised, located and used across the internet using set of standards such as SOAP, WSDL and UDDI [1]. A Web service fulfills a specific task or a set of tasks and it can be used alone or with other Web services to carry out a complex aggregation or a business transaction [1]. The widespread adoption of this technology will enable interoperability between heterogeneous platforms and help businesses to solve integration problems of their applications. The present Web service architecture is based on the interactions between three roles: service provider, service registry and service requester. The interactions among them involve publish, find and bind operations [1]. The Web service providers are growing enormously on Web resulting in Web services with similar or same functionally which has made a way for the consumers to use techniques and tools to select Web services based on their requirements. Some attempts have been made concerning the discovery of Web services based on their non-functional (i.e. what they serve) [1,3] and functional properties (i.e. how they serve) [4,5]. In Web service discovery, the matchmaking is explored through many ways such as keyword and category based [1,3], operational level description based [4], domain specific description based [6], interface signature based [5], semantic description based [7] and input, output, precondition, effect (IOPE) based [8] approaches.

The drawback of existing Web service discovery mechanism is that, it results in multiple Web services having similar or same functionality with no distinction. This caused the genesis of Web service selection problem (Fig. 1). The Web service selection is the process of choosing one Web service from many functionally similar Web services. In literature, the Web service selection is made based on personalization [9], requester's trust and connection policy [10,11], requester's past experience with the Web service [12] and the Web service quality [13,2,14].

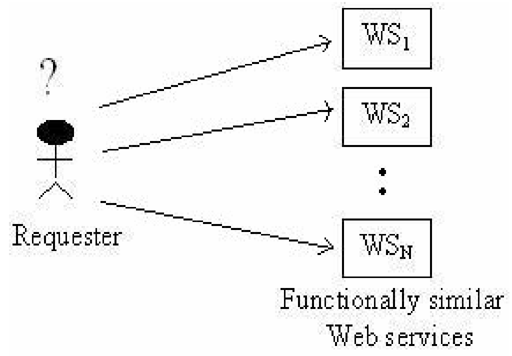


Fig. 1. Web Service Selection Problem

Quality of Service (QoS) is a measure for how well a Web service serves the requester. QoS is a crucial factor for both Web service providers and the requesters in e-business domain. QoS has an impact on the selection of quality Web service for the requester and it improves the competition among Web service providers to deliver quality services. The efforts are on to define QoS models and its impact on Web service architecture [16,15,17]. A wide range of QoS properties are proposed by number of researchers in literature. In [17], the proposed QoS model classifies QoS properties according to the point of view as system-level view, service-level view and business-level view. The paper [24] defines five major QoS properties like performance, safety, cost and interoperability, transaction support and security with sub-characteristics like response time, throughput etc. Menasce [15] defines QoS as combination of several qualities or

Demian Antony D'Mello, and Dr. V. S. Ananthanarayana are with National Institute of Technology Karnataka, Surathkal, India.

properties such as availability, security, response time and throughput. The paper [25] classifies QoS as runtime related QoS, transaction support related QoS, configuration management & cost related QoS and security related QoS. The paper [18] classifies QoS properties as general QoS properties, internet service specific QoS properties and task specific QoS properties. The paper [2] proposes extensible QoS model which classifies QoS properties into generic quality criteria and business related criteria.

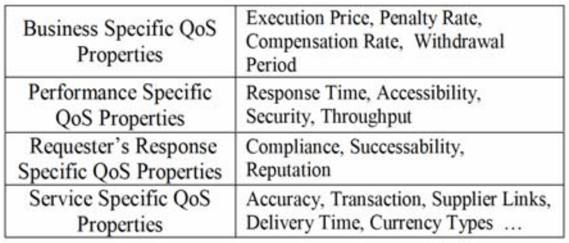
QoS can be used to select and rank functionally equivalent Web services by extending standard service oriented architecture (SOA) [18,19,20]. The Web service selection mechanism of this extended architecture ranks the Web services by matching requested QoS property values with the potential Web service QoS property values. The Web service can be selected by taking requester's average preference for each QoS property [19] without any restriction on QoS. The QoS requirement on a single QoS property (e.g. Price) can be used to filter and rank the potential Web services [21]. The Web services are also ranked based on the requester's requirements involving multiple QoS properties [2,19,13,14]. The Web services are also ranked by computing correlation (i.e. Euclidian distance) between the requested multiple QoS property values and the potential Web service QoS property values [14]. With this brief review on QoS model and QoS-aware Web service selection we conclude that, the researchers have not considered service specific QoS properties and there is well formed representation scheme (language) to specify requester's QoS requirements.

In this paper we have addressed these issues. The paper proposes QoS model for Web service selection which categorizes QoS properties based on requester's selection point of view as business specific, performance specific, requester's response specific and service specific QoS properties. We explore different types of requester's QoS requirements (i.e. constraint) and their representation schemes. We also propose a QoS based Web service selection model to find the suitable Web service for requester's QoS requirements.

The rest of the paper is organized as follows: In section 2 we define QoS model for Web service selection. Section 3 addresses different types of requester's QoS requirements. Section 4 explores the tree and XML representation schemes for requester's various QoS constraints. In section 5 we describe selection model for QoS-aware Web services. Section 6 presents the conclusion.

II. A QoS Model for Web Service Selection

QoS is a measurable non functional property of the Web service. QoS can be used to discriminate the functionally similar Web services. In this section we propose a QoS model for Web service selection which classifies QoS properties into *four* groups based on requester's selection point of view as business specific, performance specific, requester's response specific and service specific QoS properties.



Business Specific QoS Properties	Execution Price, Penalty Rate, Compensation Rate, Withdrawal Period
Performance Specific QoS Properties	Response Time, Accessibility, Security, Throughput
Requester's Response Specific QoS Properties	Compliance, Successability, Reputation
Service Specific QoS Properties	Accuracy, Transaction, Supplier Links, Delivery Time, Currency Types …

Fig. 2. A QoS Model for Web service Selection

A. Business Specific QoS Properties

The primary goal of Web service is to fulfill requester's business requirement. Business specific qualities refer to business value. We identify *four* business specific qualities namely execution price, compensation rate, withdrawal period and penalty rate.

1) *Execution Price* (E_P): Execution price is the amount of money the Web service requester has to pay to the provider for the requested service [2].

2) *Compensation Rate* (C_R)***:*** The QoS property compensation rate refers to the percentage of execution price that will be refunded when the service provider cannot honor the committed service [17].

3) *Withdrawal Period* (W_P): We define withdrawal period as the time period that commences after receipt of service request in which requester is allowed to cancel the service request without paying any penalty.

4) *Penalty Rate* (P_R)***:*** We formulate the definition of penalty rate as follows: Penalty rate is the percentage of execution price the service requester has to pay to the provider in case of service cancellation after withdrawal period.

B. Performance Specific QoS Properties

Performance specific QoS properties refer to performance of Web service system and it is the indicator of how fast the system serves the Web service request. The performance specific QoS properties dependent on static and dynamic features or behavior of Web service system. We measure the performance of Web service in terms of response time, accessibility, security and throughput.

1) *Response Time* (R_T)/*Execution Duration* (E_D): Response time (latency) is defined as the amount of time between sending a request and receiving its response [16]. Response time is the sum of service processing time and the communication delay incurred in sending a request and receiving the response. The communication delay is normally dependent on network infrastructure, network traffic and geographic location of both requester and provider.

2) *Accessibility* (A_C): We formulate the definition of accessibility as the probability that a Web service interface is ready for the access. It is the ratio of time period in which Web service interface is ready for the use over the total observed time period [17]. Accessibility of Web service normally depends in the availability of server hosting the Web service and the server throughput.

3) *Security* (S_E): Security quality can be measured based on the nature of mechanisms used for authentication,

authorization, message confidentiality, integrity, non-repudiation and resilience for denial of service attacks [17,15].

4)Throughput (T_P): We formulate the definition of throughput as the maximum number of services that a Web service system can process in a given period yielding to response.

C. Requester's Response Specific QoS Properties

We identify *three* QoS properties which are estimated based on user/requester's feedback. These qualities are measured by taking user feedback after Web service execution under the assumption that requester's are willing to give the information on service execution when asked by the feedback management mechanism and the information furnished by the requester can be trusted.

1) Compliance (C_P): Compliance of Web service refers to the ability of Web service provider to meet the service level of each QoS parameter laid out in SLA without incurring penalties [26].

2) Successability (S_C): We formulate our definition to successability as the probability that a Web service successfully completes the requested service within stipulated time interval.

3) Reputation (R_P): Reputation of a Web service is an important requester's response specific QoS property which can be measured by taking the following facts: (a) Reputation is a comparable quantity among the set of functionally similar Web services (b) Reputation of Web service dependent on frequency of service executions by the specific requester.

We formulate the definition of reputation of Web service as follows: Let M be the number of functionally similar Web services and U_1, $U_2 \ldots U_N$ be the N users who rated the Web service *WS*. The user of Web service can be asked to rate the Web service for the one time use with an integer ranging from 1 to 10. Let F be the frequency of execution of *WS* by the requester Q and R_1, $R_2 \ldots R_F$ is the corresponding ratings received by Q for execution of *WS*. The Single User Rating (*SUR*) of Web service is computed as the average of ratings received by a specific user for a given Web service. Thus *SUR* is expressed as:

$$SUR = \frac{R_1 + R_2 + \ldots + R_F}{F}$$

The reputation of Web service is the average of *SUR* of N requesters. Thus reputation (R_P) is estimated as:

$$R_P = \frac{SUR\ of\ U_1 + SUR\ of\ U_2 + \ldots + SUR\ of\ U_N}{N}$$

D. Service Specific Qualities

Service specific qualities are dependant on the type of Web service and the service domain. These qualities may vary service to service. In this paper we identify accuracy, transaction property and a few domain specific QoS properties. The accuracy QoS property is specific to Web service involving numerical results and the transaction property is specific Web services involving execution of business transactions (short or long living transactions).

1) Accuracy (A_R): Accuracy is defined as the accuracy of results in a numerical manner [17].

2) Transaction (T_S): Transaction property is used to maintain data consistency (use of undo procedure, duration of undo procedure) [2]. Transaction property describes the ACID properties of Web service transactions.

We can also define QoS property *Delivery Time* (the delay incurs in delivering the purchased product/goods) for purchase related Web services in shopping and transportation domain. The QoS property *Currency Types* (number of currencies available for conversion) can be defined for currency conversion service in financial services domain.

The important feature of all QoS properties is *that from requester's pint of view they are either positive or negative in nature*. For positive QoS property, a higher value indicates the better quality and for the negative QoS property higher the values lower the quality. For example QoS property R_T is negative since lesser value of R_T indicates higher quality and R_P is positive as higher value indicates reputed Web service.

III. Requester's QoS Constraints

The Web service requester normally expects some requirements on QoS properties to be satisfied by the Web services. We define QoS constraint as the Web service requester's requirement on some QoS properties. Formally, QoS constraint is a relational expression defined on some QoS properties. Normally QoS constraints are different for individual requesters. For example, consider online buying scenario in shopping domain where the requester normally looks for a Web service with minimum price and that supports quick delivery but some buyers focus only on quick delivery and others concentrate on minimum Web service price (delivery price). Thus Web service requesters can have different requirements on several QoS properties with varied preferences.

A. Requester's QoS Constraints

We categorize QoS constraints based on QoS constraint structure as *simple* and *composite* QoS constraints. A simple QoS constraint normally deals with one QoS property. A simple QoS constraint takes the following format: Q_i *cp* V_i where Q_i refers to QoS property, *cp* refers to comparison operator ($<$, $>$, $\leq$, $=$, $\neq$ and $\geq$) or membership operator (*in*) and V_i refers to expected single value or range of values for Q_i.

We further classify simple QoS constraint as *point, implicit range* and *explicit range* QoS constraints based on the nature of *cp*. A simple QoS constraint with equality operator (=) is called as point QoS constraint. For example the buyer might say "I need a book seller who delivers book in one day". This is point constraint and can be written as "delivery time = 1". A simple QoS constraint with comparison operators $<$, $>$, $\leq$, $\neq$ and $\geq$ is referred as implicit range QoS constraint. In implicit range QoS constraint the minimum value for Q_i or maximum value for Q_i or both minimum and maximum value for Q_i is implicit. For example buyer might say "I need book seller with reputation more than 7 (out of ten)". This is implicit range QoS constraint which can be written as "reputation > 7". Here maximum value for reputation is 10 (according to the QoS model). Thus the range is 8-10 where value 10 is implicit. A simple QoS constraint with explicit lower and upper bounds for Q_i (QoS constraint with membership operator *in*) is referred as explicit range QoS constraint. For example buyer might say "I need a book seller whose delivery price is between 4 to 8 dollars". This is explicit range QoS constraint which can be written as "price *in* [4-8]".

Composite QoS constraint is composed of multiple simple QoS constraints using constraint composition operators *AND* and *OR*. For example buyer might say "I am interested in book seller with price less than $3 and delivery time less than 3 days". This is composite QoS constraint which can be represented as "price < 3 *AND* delivery time < 3". A composite QoS constraint takes the form C_1 *op* C_2 *op* C_3 *op* …*op* C_P where C_i refers to simple QoS constraint and *op* denotes constraint composition operator *AND* or *OR*. In general a QoS constraint takes the general form $(Q_i\ cp\ V_i)\ (op\ (Q_j\ cp\ V_j))^*$. The Web service requester can enforce either simple or composite QoS constraints during Web service selection to choose desired Web service.

B. Requester's QoS Constraint Preferences

Consider the book buying scenario; the book buyers will be having several requirements on the seller's quality. Some buyers prefer express delivery and others for the minimum delivery price (service price). There are buyers who prefer both speedy delivery and cheaper service with more preference (weight) to price. Consider the buyer QoS requirements as follows: (a) book seller service that delivers book within 8 days and delivery price < $20 and penalty rate < $5 with varied preferences to delivery time, price and penalty. (b) book seller service with reputation > 6 (out of ten) and delivery price < $25 with equal preference to price and reputation. The buyer expects one of the requirements to be satisfied by the book seller and he gives more preference to QoS requirement (a) over QoS requirement (b). If the Web service requester has several requirements on different qualities with varying preference to each requirement, then we need to identify the mechanism to represent requester's such QoS requirements.

C. Requester's Alternative QoS Constraints

Consider again the book buying scenario; it is common tendency of the buyer to enforce strong requirements on seller's QoS in shopping domain. It is likely that buyer may not find the seller for his strong QoS demands (requirements) which makes him to enforce slight weak requirements. As an example consider the buyer's strong and weak QoS requirements as follows: (1) book seller Web service that delivers book within a day with a delivery price less than $3. (2) book seller Web service that delivers book within 3 days with a delivery price less than $5 and has reputation above 6 (out of ten). If the requester has a set of alternative (substitute or choice based) requirements on QoS in the order of diminishing preferences then we need a mechanism to represent requester's alternative QoS requirements.

IV. REQUESTER'S QOS CONSTRAINT MODELING

Consider the requester's composite QoS constraint involving different QoS properties (as defined in section 2) with varied preferences. Here we propose the tree and an XML representation schemes for requester's QoS constraints.

A. Tree Structure for Requester's QoS Constraints

In this section we explore the novel tree structure to represent requester's QoS constraints with varied preferences.

1) AND-OR Tree: An AND-OR Tree is a non empty rooted tree of order N, with finite number of nodes and each node can contain *zero* or *two* or N ($N > 2$) child nodes. A node with no child is called as a leaf node and node with C, ($2 \leq C \leq N$) child nodes is referred as internal node. The internal node contains one item of information i.e. AND or OR and the leaf node contains finite number of items of information.

The important property of AND-OR tree is that, all the leaf nodes will be at the same level i.e. we assign level 0 to all the nodes with no child and the level of any internal node (say X) can be computed as maximum among the level of children nodes + 1. The level of any AND-OR tree node can be computed recursively as follows:

- level(X) = 0, if X is a leaf
- level(X) = Max{level(Y_1),... , level(Y_C)} + 1, if X is a internal node and the nodes Y_1 to Y_C are the C children of node X.

Let H be the height of an AND-OR Tree, then the level of root node will be H and the levels of internal nodes (except root) will be between 1 and H-1. Fig. 3 illustrates the property of an AND-OR tree. The nodes with labels G, F and E are internal nodes and the nodes A, B, C and D are leaves with finite information items. The node G represents the root which of type *AND* and the node F is of type *OR*. The height of a tree is 3 thus the level of root node G is 3 and the level of node F is calculated as the maximum of level of node B and node E i.e., max (0,1) + 1 = 2.

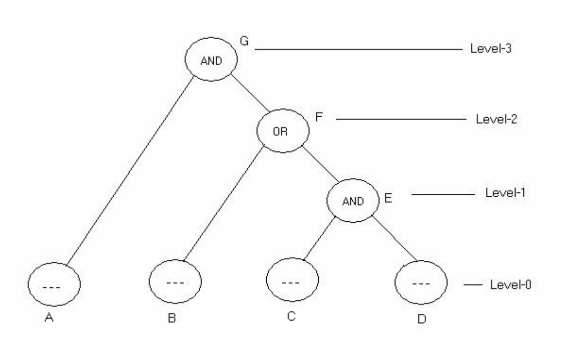


Fig. 2. An AND-OR Tree

2) Weighted AND-OR Tree: A Weighted AND-OR tree is a AND-OR tree where every edge between parent and child node is labeled with a non-negative real value (weight) in an interval (0, 1) such that for any parent node the sum of edge labels of all child nodes is equal to one i.e. for any parent node P with C ($2 \le C \le N$) child nodes the sum of edge weights W_{PCi} ($1 \le i \le C$) is equal to 1.

3) Quality Constraint Tree (QCT): A Quality Constraint tree is a Weighted AND-OR tree whose leaf node contains either three or four information items. A leaf node contains QoS property (Q_i), comparison or membership operator (cp) and the expected QoS property value (V_i) or lower and upper bound values (V_{il} & V_{iu}) for Q_i. The internal node refers to constraint composition operator op. The label W_{XY} on the edge between any two nodes X and Y represents the preference for sub-tree rooted at Y while traversing from root to leaf i.e. the edge label represents requester's preference to either simple or composite QoS constraint defined for the sub-tree rooted at node Y. Thus leaf node represents simple QoS constraint and any sub-tree rooted at internal node represents composite QoS constraint.

The requester's QoS constraint can be represented using QC tree (QCT). Consider the requester's QoS constraints as follows: (a) Response Time (R_T) < 24 *AND* Price (E_P) < \$4 with preference to each simple QoS constraint 0.6 and 0.4 respectively. (b) Reputation (R_P) > 7 (out of 10) *AND* Price (E_P) < \$6 with equal preference (i.e. 0.5) to each simple QoS constraint. Assume that requester gives equal preference to both QoS constraint (a) & QoS constraint (b). Fig. 4 shows the QC tree for requester's QoS constraints. The nodes *D*, *E*, *F* and *G* are found at level zero. The nodes *B* and *C* are found at level one. The node *A* (root) is found at level two. Thus the height of QC tree is 2. The edge label refers to requester's preference for QoS constraints. For example edge label '0.5' on the edge between node *A* and node *B* refers to requester's preference for QoS constraint (a).

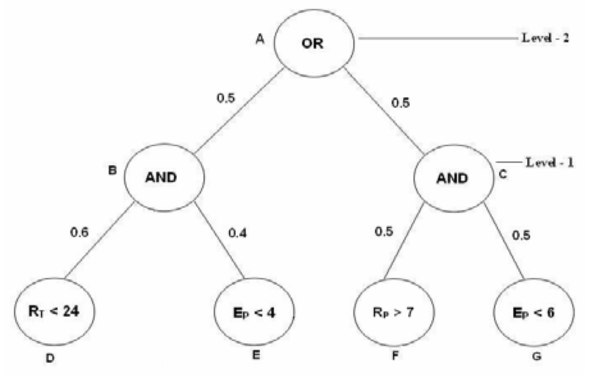
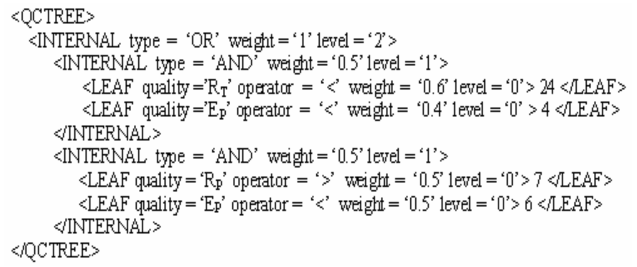


Fig. 4. QC tree for Requester's QoS constraints

B. An XML Representation for QoS Constraints

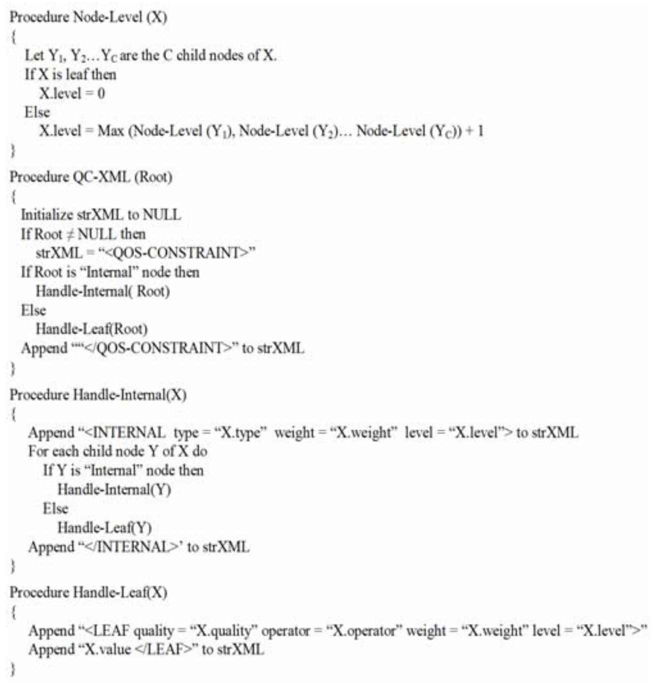
Here we present an XML representation for requester's QoS constraint i.e. an XML representation for QC tree. An XML representation of QCT can be used to embed requester's QoS requirements within the SOAP message for QoS-aware Web service selection. An XML representation also facilitates the use of existing XML parsers during Web service selection mechanism. In XML representation, the QC tree is represented using a tag <QCTREE> with sub-tags <INTERNAL> and <LEAF>. The internal node is represented with a tag <INTERNAL> and it takes three attributes namely *type*, *weight* and *level*. For any internal node *X*, the *type* attribute refers to type of internal node (AND/OR), the *weight* attribute refers to label on the edge between node *X* and node *P* where *P* is proper (strict) ancestor of *X* and the *level* attribute refers to level of the internal node. The leaf node is represented using tag <LEAF> that takes three attributes namely *quality*, *operator*, *weight* and *level*. For any leaf node *Y*, the *quality* attribute refers to QoS property Q_i, *operator* attribute refers the comparison operator *cp*, the *weight* refers to label on the edge between node *Y* and node *P*, where node *P* is proper ancestor of *Y* and the *level* attribute refers to level of the leaf node. The root refers to whole QoS constraint thus, its weight attribute of is set to 1. Fig. 5 shows an XML representation of QC tree of Fig.4.



```
<QCTREE>
  <INTERNAL type = 'OR' weight = '1' level = '2'>
     <INTERNAL type = 'AND' weight = '0.5' level = '1'>
        <LEAF quality ='R_T' operator = '<' weight = '0.6' level = '0'> 24 </LEAF>
        <LEAF quality ='E_P' operator = '<' weight = '0.4' level = '0' > 4 </LEAF>
     </INTERNAL>
     <INTERNAL type = 'AND' weight = '0.5' level = '1'>
        <LEAF quality = 'R_P' operator = '>' weight = '0.5' level = '0'> 7 </LEAF>
        <LEAF quality = 'E_P' operator = '<' weight = '0.5' level = '0'> 6 </LEAF>
     </INTERNAL>
</QCTREE>
```

Fig. 5. An XML Representation for QCT

Fig. 6 depicts the procedures which transforms QC tree to an XML representation. The procedure *Node-Level* computes the level of each node of QC tree and the procedure *QC-XML* handles the entire QC tree. The procedures Handle-Internal and Handle-Leaf) are repeatedly called by the *QC-XML* to complete QC tree transformation.



```
Procedure Node-Level (X)
{
  Let $Y_1$, $Y_2$…$Y_C$ are the C child nodes of X.
  If X is leaf then
    X.level = 0
  Else
    X.level = Max (Node-Level ($Y_1$), Node-Level ($Y_2$)… Node-Level ($Y_C$)) + 1
}
Procedure QC-XML (Root)
{
  Initialize strXML to NULL
  If Root $\neq$ NULL then
    strXML = "<QOS-CONSTRAINT>"
  If Root is "Internal" node then
    Handle-Internal( Root)
  Else
    Handle-Leaf(Root)
  Append ""</QOS-CONSTRAINT>" to strXML
}
Procedure Handle-Internal(X)
{
   Append "<INTERNAL type = "X.type" weight = "X.weight" level = "X.level"> to strXML
   For each child node Y of X do
     If Y is "Internal" node then
       Handle-Internal(Y)
     Else
       Handle-Leaf(Y)
   Append "</INTERNAL>' to strXML
}
Procedure Handle-Leaf(X)
{
   Append "<LEAF quality = "X.quality" operator = "X.operator" weight = "X.weight" level = "X.level">"
   Append "X.value </LEAF>" to strXML
}
```

Fig. 6. Procedures for QC tree to XML transformation

C. Tree Representation for Alternative QoS Constraints

We extend the QC tree structure to represent requester's set of alternative QoS constraints with diminishing preferences. We propose the concept of extended QC tree to represent requester's alternative QoS constraints. Consider $S = \{QC_1, QC_2 \ldots QC_R\}$ is the set of R QoS constraints of the requester in the order of diminishing preferences. Let QC_1 be the strong QoS constraint and QC_2 to QC_R be alternative QoS constraints in the order of diminishing preferences. Let QCT_1, QCT_2 … QCT_R be QC trees for R alternative QoS constraints QC_1, $QC_2 \ldots QC_R$. We create a new node called XOR and then we attach R QC trees to the XOR node. Now the XOR node becomes the root for all QC trees. The sub-tree rooted at child nodes of XOR node refers to requester's alternative QoS constraints. The weight for the edge between XOR node and any QC tree root is assigned as follows. A real number in an interval (0,1) is assigned to each edge such that the sum of weights of edges between XOR node and root of QC trees is equal to 1. Let E_1, $E_2 \ldots E_R$ be the edges between XOR node and roots of R QC trees. The weight (W_i) on the edge E_i is calculated as $2*(R\text{-}i+1)/R(R+1)$. The extended QC tree for the requester's alternative QoS constraints is depicted in Fig. 7. The buyer's strong QoS requirement is: $E_D = 1$ AND $E_P < 3$. The buyer's weak QoS requirement is: $E_P < 5$ AND $E_D < 3$ AND $R_P > 6$. Here we assume that the buyer gives an equal preference to E_P and E_D in strong QoS constraint. The buyer also gives preference 0.3, 0.5, 0.2 to E_P, E_D and R_P in weak QoS constraint. The strong QoS constraint gets higher weight (preference) than the weak QoS constraint. The sub-tree rooted at B represents weak QoS constraint and the sub-tree rooted at C represents strong QoS constraint.

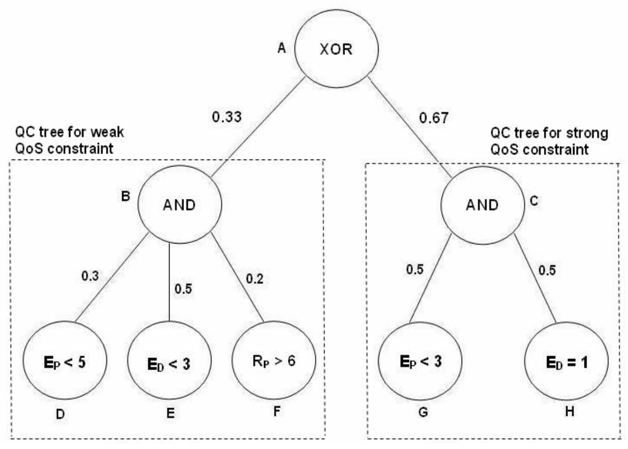
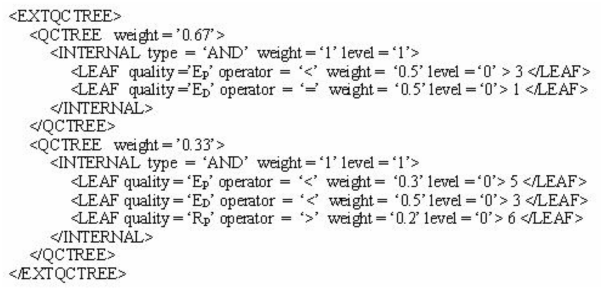


Fig. 7. Extended QC tree for Buyer's Alternative QoS Constraints

D. An XML Representation for Alternative QoS Constraints

Here we present an XML representation for requester's alternative QoS constraints. In an XML representation, the requester's alternative QoS constraint set is represented with a tag <EXTQCTREE> and multiple sub-tags <QCTREE>. The tag <QCTREE> refers to QC tree with attribute *weight* and sub-tags <INTERNAL> and <LEAF>. Fig. 8 shows an XML representation of extended QC tree for the buyer's set of alternative QoS constraints. The QC tree to XML transformation procedure can be used with minor modifications to transform an extended QC tree to an XML representation.



```
<EXTQCTREE>
  <QCTREE  weight = '0.67'>
    <INTERNAL type = 'AND' weight = '1' level = '1'>
      <LEAF  quality ='EP' operator = '<'  weight = '0.5' level = '0' > 3 </LEAF>
      <LEAF  quality ='ED' operator = '='  weight = '0.5' level = '0'> 1 </LEAF>
    </INTERNAL>
  </QCTREE>
  <QCTREE  weight = '0.33'>
    <INTERNAL type = 'AND' weight = '1' level = '1'>
      <LEAF quality = 'EP' operator = '<'  weight = '0.3' level = '0'> 5 </LEAF>
      <LEAF quality = 'ED' operator = '<'  weight = '0.5' level = '0'> 3 </LEAF>
      <LEAF quality = 'RP' operator = '>'  weight = '0.2' level = '0'> 6 </LEAF>
    </INTERNAL>
  </QCTREE>
</EXTQCTREE>
```

Fig. 8. An XML Structure for Buyer's Alternative QoS Constraints

V. QoS-aware Web service selection model

We propose the QoS broker based Web service selection model with an objective of selecting the best (most suitable) Web service that satisfies requester's QoS constraints. The proposed model (Fig. 9) has five roles namely Web service consumer, Web service provider, broker, QoS certifier and the QoS-aware UDDI registry. The model adapts QoS vocabulary as described in section 2.

A. The Roles and Interactions

The Web service provider offers service by publishing the Web service into the QoS-aware UDDI registry; the Web service requester needs a service offered by the provider; the QoS-aware UDDI registry is a repository for registered Web services with search options; the QoS certifier role is to verify Web service provider's QoS claims as described in [25]; the new role broker is to select desired Web service for the Web service requester. The broker also records the feedback of legitimate requester after service invokation. The QoS-aware UDDI registry uses the extended UDDI data model as described in [25]. The publication of Web service & its QoS details is as proposed in [25]. The new role broker can be implemented as a Web service with the following functionalities. The broker obtains feedback from requesters after service use in order to estimate the requester's response specific QoS properties like reputation, successability and compliance. The collection and calculation of QoS ratings from the requester can be done as described in [22]. The broker accepts SOAP request message for QoS-aware Web service selection from the requester's selection tool and then extracts the QoS constraint information (an XML equivalent of QC tree) from the SOAP message header. Now the broker constructs a new SOAP request for Web service discovery with QoS by sending only the service name and desired QoS properties to the QoS-aware UDDI registry. A sample SOAP request of the requester's selection tool and the broker for the requester's QoS constraints is shown in Fig. 10. The requester's QoS constraints are: (a) $E_D < 24$ *AND* $E_P < 8$ with preference 0.6 and 0.4 to E_D and E_P. (b) $E_P < 4$. The requester wants one Web service that satisfies either QoS constraint (a) or QoS constraint (b) with same priority to both the QoS constraints.

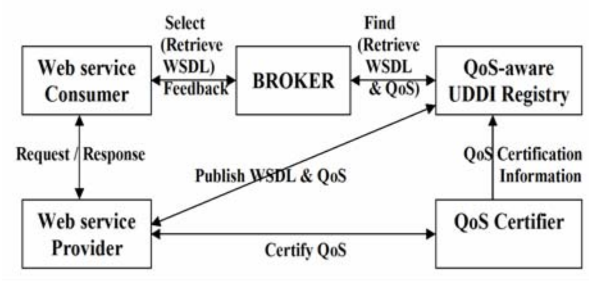


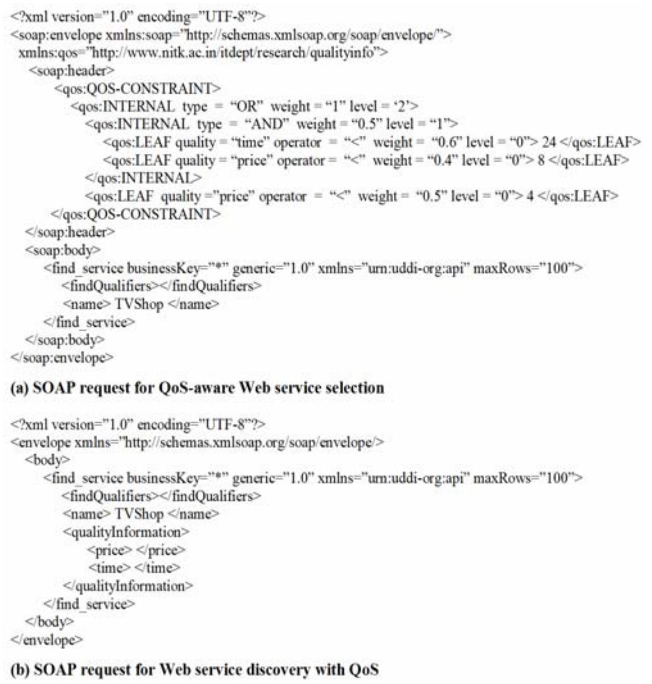
Fig. 9. A model for QoS-aware Web service selection

The interaction among different roles of the new Web service selection model is summarized below:

1. Provider publishes business entity details and obtains business key and authentication information from the QoS-aware UDDI registry for further transactions
2. Provider sends the QoS details along with the authentication information to the QoS certifier for certification
3. After certification, the QoS certifier sends the certification id to the provider and to the QoS-aware UDDI registry
4. Now the provider publishes Web service details and QoS with QoS-aware UDDI registry
5. The requester uses the tool to specify his QoS constraints with requested functionality and the tool sends a SOAP request message for Web service selection to the broker
6. The broker extracts QoS constraints from the header and then constructs a new SOAP request message for the Web service discovery with QoS
7. QoS-aware UDDI registry sends a response having discovered Web services with QoS information
8. Now broker extracts QoS and Web service information from the message. The broker now retrieves requester's response sensitive QoS information like compliance & reputation from local store (if required) and executes the selection algorithm
9. After execution of selection algorithm, broker sends the best Web service (QoS optimal) to the requester through the selection tool.

B. The Web Service Selection mechanism

The broker executes the selection algorithm for the requester's QoS constraints as follows: The algorithm takes QC tree *T* of height *H* and the candidate Web services with QoS information (Functionally similar Web services returned by the UDDI registry based on keyword matching) as an input and results in a single Web service which is most suitable for the requester based on requester's QoS constraints and preferences. The algorithm traverses QC tree in level order fashion (level 0 to level H) and treats leaf and internal nodes in a different manner. At leaf nodes algorithm performs the following *three* actions: (1) *Filtering* (2) *Scaling* and (3) *Ranking*. In filtering phase, the Web services that satisfy simple QoS constraint defined at the leaf node are selected. The scaling phase normalizes the QoS values of selected Web services to a non-negative real valued number in an interval [0,1] using min-max normalization technique [14]; where the higher normalized values represent higher level of quality. In ranking phase normalized values are multiplied with the weight (preference given to the QoS constraint) to get new values representing scores for the Web services. At internal nodes, the algorithm performs *two* actions namely *Filtering* and *Ranking* which are dependent on the type of node (AND/OR). In filtering phase, if the node is *AND* then the Web service present in *all* its child nodes is selected. If the node is *OR* then *distinct* Web services in the descending order of their scores are selected from its child nodes. In ranking phase, if the node is *AND* then the score of selected Web service is computed as the sum of scores of selected Web service at its child nodes multiplied with the weight of sub-tree rooted at *AND* node. If the node is *OR* then the score of selected Web service is multiplied with the weight of sub-tree rooted at *OR* node. After ranking Web services at the root node, the Web services are sorted in the descending order of their score. Finally the algorithm returns first Web service to a requester as best (most suitable) Web service. The requester is also allowed to specify his alterative QoS constraints as discussed in section 3. In such scenario, the selection algorithm handles each QC tree attached to the root (XOR node) in the order of preference until suitable Web service is found for the requester.



```
<?xml version="1.0" encoding="UTF-8"?>
<soap:envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 xmlns:qos="http://www.nitk.ac.in/itdept/research/qualityinfo">
   <soap:header>
       <qos:QOS-CONSTRAINT>
          <qos:INTERNAL type = "OR" weight = "1" level = '2'>
             <qos:INTERNAL type = "AND" weight = "0.5" level = "1">
                <qos:LEAF quality = "time" operator = "<" weight = "0.6" level = "0"> 24 </qos:LEAF>
                <qos:LEAF quality = "price" operator = "<" weight = "0.4" level = "0"> 8 </qos:LEAF>
             </qos:INTERNAL>
             <qos:LEAF quality ="price" operator = "<" weight = "0.5" level = "0"> 4 </qos:LEAF>
       </qos:QOS-CONSTRAINT>
   </soap:header>
   <soap:body>
      <find_service businessKey="*" generic="1.0" xmlns="urn:uddi-org:api" maxRows="100">
         <findQualifiers></findQualifiers>
         <name> TVShop </name>
      </find_service>
   </soap:body>
</soap:envelope>
```

(a) SOAP request for QoS-aware Web service selection

```
<?xml version="1.0" encoding="UTF-8"?>
<envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/>
   <body>
      <find_service businessKey="*" generic="1.0" xmlns="urn:uddi-org:api" maxRows="100">
         <findQualifiers></findQualifiers>
         <name> TVShop </name>
         <qualityInformation>
              <price> </price>
              <time> </time>
         </qualityInformation>
      </find_service>
   </body>
</envelope>
```

(b) SOAP request for Web service discovery with QoS

Fig. 10. A SOAP request message with QoS Constraint Information

VI. Conclusion

The paper proposes the QoS model for Web service selection which classifies QoS properties based on requester's selection point of view. We explored the different types of requester's QoS constraints with illustrations. We presented a tree structure and an XML representation for requester's QoS constraints with varied preferences. The also paper explores representation schemes for requester's alternative (substitute) QoS constraints. The paper proposes QoS-aware Web service selection model which finds the suitable Web service from functionally similar Web services based on requester's QoS constraints and preferences.

VII. References

[1] H. Kreger: Web Services Conceptual Architecture (WSCA 1.0), Published in May 2001, Available: www.ibm.com/software/solutions/ webservices/ pdf/wsca.pdf.

[2] Y. Liu, A. H. H. Ngu, and L. Zeng: QoS Computation and Policing in Dynamic Web Service Selection, Proceedings of the WWW 2004, ACM 1-58113-912-8/04/0005, pp. 66-73, 2004.

[3] UDDI Technical White Paper, Published on 2000, Available: www. uddi.org/ pubs/ Iru_UDDI_Technical_White_Paper.pdf.

[4] Z. Shen and J. Su: Web Service Discovery Based on Behavior Signatures, Proceedings of the IEEE International Conference on Services Computing 2005, IEEE 2005.

[5] Y. Wang and E. Stroulia: Flexible Interface Matching for Web-Service Discovery, Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE'03), IEEE, 2003.

[6] D. Rocco and J. Caverlee: Domain-specific Web Service Discovery with Service Class Descriptions, Proceedings of the IEEE International Conference on Web Services (ICWS'05), IEEE 2005.

[7] S. B. Mokhtar, A. Kaul, N. Georgantas, and V. Issarny: Towards Efficient Matching of Semantic Web Service Capabilities, Proceedings of the International Workshop on Web Services Modeling and Testing (WS-Mate 2006).

[8] X. Gao, J. Yang, and M. P. Papazoglou: The Capability Matching of Web Services, ISBN 0-7695-1857-5/02, IEEE, 2002

[9] W. T, Balke, and M. Wagner: Towards Personalized Selection of Web Services, Proceedings of the WWW 2003, ISBN 963-311-355-5, 2003.

[10] S. Ali, S. A. Ludwig, and O. F. Rana: A Cognitive Trust-based Approach for Web Service Discovery and Selection, Proceedings of the Third European Conference on Web Services (ECOWS'05), IEEE 2005.

[11] M. Marchi, A Mileo, and A. Provetti: Declarative Policies for Web Service Selection, Proceedings of the IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'05), IEEE 2005.

[12] Julian Day: Selecting the Best Web Service, Availbale: http://bistrica.usask.ca/ MADMUC/Pubs/day880.pdf.

[13] L. H. Vu, M. Hauswirth, and K. Aberer: QoS Based Service Selection and Ranking with Trust and Reputation Management, Proceedings of the CoopIS/DOA/ODBASE 2005, LNCS 3760, pp. 466-483, 2005.

[14] L. Taher, R. Basha and H. E. Khatib: Establishing Association between QoS Properties in Service Oriented Architecture, Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP'05). IEEE 2005.

[15] D. A. Menasce: QoS Issues in Web Services, IEEE Internet Computing, pp. 72-75, Nov-Dec 2002, IEEE.

[16] Mani and A. Nagarajan: Understanding quality of service for Web services, Available: www.emeraldinsight.com/Insight/html/Output/ Published/EmeraldFullTextArticle/Pdf/1720140505.pdf, Published in 2002.

[17] G. Yeom, T. Yun and D. Min: A QoS Model and Testing Mechanism for Quality-driven Web Services Selection, Proceedings of the Second International Workshop on Collaborative Computing, Integration, and Assurance (SES-WCCIA'06), IEEE 2006.

[18] M. A. Serhani, R. Dssouli, A. Hafid, and H. Sahraoui: A QoS broker based architecture for efficient web service selection, Proceedings of the IEEE International Conference on Web Services (ICWS'05), IEEE 2005.

[19] J. Hu, C. Guo, H. Wang, and P. Zou: Quality Driven Web Services Selection: Proceedings of the IEEE International Conference on e-Business Engineering (ICEBE'05), IEEE 2005.

[20] D. Z. G. Garcia, M. B. F Toledo: A Web Service Architecture Providing QoS Management, Proceedings of the Fourth Latin American Web Congress (LA-Web '06), IEEE 2006.

[21] M. Kerrigan: Web Service Selection mechanisms in the Web Service Execution Environment (WSMX), Proceedings of the SAC'06, ACM 1-59593-108-2/06/0004, pp. 1664-1668, 2006.

[22] V. Deora, J. Shao, W. A. Gray, and N. J. Fiddian: A Quality of Service management Framework Based on User Expectations, Proceedings of the ICSOC 2003,. LNCS 2910, pp. 104-114, 2003.

[23] SAP UDDI V3 Test Public Business Registry: Available http://udditest.sap.com/webdynpro/dispatcher/sap.com/tc~uddi~webui~ wdp/UDDIWebUI/, visited on April 2007.

[24] Y. J. Seo, H. Y. Jeong and Y. J. Song: Best Web Service Selection Based on the Decision Making Between QoS Criteria of Service, ICESS 2005, LNCS 3820, pp. 408-419.

[25] Shuping Ran: A Model for Web Services Discovery With QoS, ACM 2003.

[26] K. Sravanthi, K. Shonali and S. W. Loke: Reputation= f(User Ranking, Compliance, Verity), Proceedings of the IEEE International Conference on Web Services (ICWS'04), IEEE 2004.

VIII. Biographies

Demian Antony D'Mello received his Bachelor degree in Computer Engineering in 1999 from Mangalore University, India and his Master degree in Computer Science and Engineering in 2003 from National Institute of Technology Karnataka, Surathkal, India. He is working in the Department of Computer Science and Engineering, St. Joseph Engineering College, Mangalore, India since 2003. At present he is a research student in the Department of Information Technology, National Institute of Technology Karnataka, Surathkal, India. His research interests are in the areas of Web services and image processing.

Dr. Ananthanarayana V.S. received his Bachelor degree in Computer Science and Engineering in 1990 from Karnatak University, India, his Master degree in Computer Science and Engineering in 1995 and his Doctoral degree in 2001 from Indian Institute of Science, Bangalore, India, and Post Doctoral Fellow in 2005 from Memorial University of Newfoundland, Canada. He is currently a professor of Department of Information Technology, National Institute of Technology Karnataka, Surathkal, India. His research interests are in the areas of database systems, data mining, distributed computing and Web services.