A Study on Quality Broker to Users Web Service Selection Based on Non-Functional Attributes

Sung-Ho Sim *, Young-Jae Song
Dept. of Computer Engineering
KyungHee University, YongIn, Korea

ABSTRACT

The recent Web service field emerges as the fastest growing IT paradigm as a result of the increasing interest in SOA (Services-Oriented Architecture) and the expansion of B2B market. With an increasing number of Web service that provide similar features, it becomes more important to provide the most appropriate service for the user's request. A service user in general requires the quality information of Web service when selecting a service among a number of similar Web services. Yet, finding a service through UDDI (Universal Description, Discovery and Integration) does not consider the non-functional aspects of users because it is only based on the functional aspects. That is, the quality, non-functional aspect will be an important factor for the mutual success of the user and provider. Using 3 factors in the Qos factors of the existing studies: the execution cost, reliability, and the quality level, the QoS of Web service is saved and the factors for the QoS are recorded in order to consider non-functional factors when selecting a Web service in this study. The Quality Broker determines the rank and shows the desired result of the service for users. The Quality Broker suggested in this thesis can be used to select a Web Service that considers the user-oriented and non-functional factors.

***Keywords**: Web Service, Quality Broker, QoS, non-functional attribute*

1. INTRODUCTION

The recent Web service is popular as the base of the next-generation e-business to solve the problem of integrated service environment within and between enterprises. About 30% of the current system uses network system and many of the other systems are used in closed environment. More studies are required to make up the current problems caused by a number of users for the future of Web service even though there are many studies and researches going on at present.[1], [2]

A Web service is an application designed to communicate over HTTP using the XML form such as SOAP (Simple Object Access Protocol). [3], [4] Web service defines interface with WSDL (Web Service Description Language) which explains the Web service of the service provider and uses UDDI (Universal Description, Discovery and Integration Protocol) to seek a new service. [5], [6]

The base architecture of Web service consists of Service Provider, Service Consumer, and UDDI and UDDI only provides the functional aspects of the service when searching a Web service using UDDI. [10]

However, there are not many Web services that can be used even though a number of Web services are registered in UDDI. Also, WSDL only provides the interface of a service and does not consider the non-functional factors.[1] Such problems cause an additional cost and increased service processing time when using a Web service as well as the reliability problem for the service. And it is also possible that it is unable to use the service after completing the design for implementation and determining a service.[12] Therefore, there have been studies such as Web Service Broker, QoS Registry, and WS-QoS XML Schema to solve such problems. [7]-[9]

The existing studies show that the functional aspect of Web service uses the existing architecture that is expanded in the non-functional aspect of Web service. Although services are selected using the Broker considering the non-functional aspect with the XML Schema for QoS, it has a weak point that might cause burden on the server due to the complexity of XML Schema and matching between user and provider and the delay of performance time via the Broker. This study researches the method of Web service selection that considers the non-functional aspect of the user when selecting a Web service by proposing Quality Broker.

2. RELATED WORKS

This chapter describes WS-QoS XML Schema and the expandable QoS model, general QoS factor, and business-related factor.[7]

This is an excellent paper selected from the papers presented at ICCC 2008.

* *Corresponding author. E-mail : shim5800@khu.ac.kr*

Manuscript received Mar. 03, 2009 ; accepted Apr. 16, 2009

2.1 WS-QoS XML Schema

The WS-QoS XML Schema is proposed for the QoS-oriented service description. The WS-QoS architecture is based on the XML Schema. [7]

There are 3 kinds of WS-QoS XML documents: QoSRequirementDefinition that descrbes the QoS requirement of the user, QoSRequirementDefinition that includes the provider's QoS description, and WSOntology that defines the appropriate QoS factor and protocol reference. WSQoSRequirementDefinition and WSQoSOfferDefinition are of the tQoSDefinition type and include more than one QoSInfo. Fig. 1 shows the general type WS-QoS document.

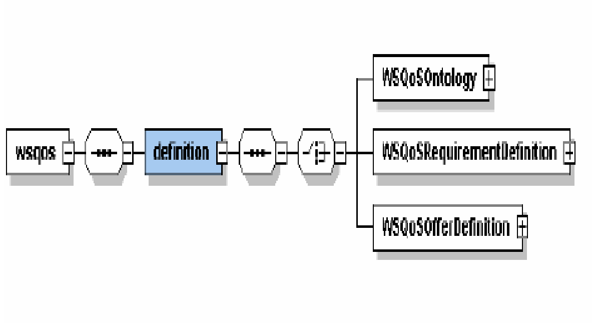


Fig. 1. Type of WS-QoS document

WS-QoS XML Schema, in general, each contains QoSRequirementDefinition. And WSQoSOfferDefinition contains defaultQoSInfo. defaultQoSInfo composed with protocol which requires for performance of server, information level of QoS compliance to transmitting and security for transaction. serverQoSMetrics defines server QoS from defaultQoSInfo such as data processing time, times of response per second from trust, and possibility. Next, transportQoSPriorities defines delay of transmitting, Jeter, Process amount and packet damages through the network transmitting. Finally, SEcurityAndTransaction defines the protocol which needs for successful execution.In QoSInfo

In QoSInfo, customMetric, customPriority, Protocol refer to WSQoSOntoloty's meticDefinition, protocolDefinition, priorityDefinition WSQoSrequirementDefinition write out by one who request and WSQoSOfferDefinition write out by provider. WSQoSOfferDefinition is provided reference to WSDL file and the adjustments are possible without any changes from WSDL file.

The purpose of WE-QoS XML is architecture for the use of clear Web service that goes with the QoS. That is, it is the method to select the most optimizing Web service by describing and matching the requirements of both the user and provider. A weak point of the WS-QoS XML Schema is the complexity of the XML Schema and difficulty of matching the user and provider.

2.2. Expandable QoS Model

The factors to process QoS of the existing Web services are the cost, execution time, reliability, and possibility of use. [8] Such general factors are not good enough to be widely applied in other domains. QoS must include specific factors of domain and expandability. Besides, most of the current methods provide the interface to access the QoS information which is affected by the service provider and depend on the service broker that gives the QoS information. The service providers cannot give away their QoS information and the method to collect the QoS information exclusively by monitoring carries a load because it requires to have a number of Web services to be searched. And other approaching methods (Recommend a special service provider or use a third party that determines the QoS information) are costly and not flexible. The expandable QoS model framework which is based on the execution monitoring can be determined by the users and collected by the service provider.[11]

2.3. General QoS Factors

Three general QoS factors to determine the service are the execution cost, execution time, and reputation.

2.3.1. Execution Cost: The execution cost is a cost that the service user has to pay to the service provider for the use of Web service such as the credit evaluation. The execution cost to use the Web service is $q_{pr}(s)$.

2.3.2. Execution Time: The execution time is the expected processing time from sending a request to the response of the service. The execution time is the sum of the processing time and transmitting time $T_{porcess}(S)$ and transmitting time $T_{trans}(S)$. The execution time is obtained by monitoring.

2.3.3. Reputation: The reputation of service is the evaluation of the service's reliability. It is evaluated depending on the user's experience. Each user has different opinions for the same service. The value of reputation is defined by the average ranking given to be served for the users. It is the user's ranking for the service reputation. n is the number that the service is evaluated. Normally, the service user is given the range for the ranking of Web service. For example, Amazon.com has the range of [0, 5].

2.4. Factors related to the business

The factors related to the business vary according to different domains. According to different domains, the QoS factors can be expanded and add QoS factors appropriate for the application. In the application for the expandable QoS model, the utility is expanded in three factors.

2.4.1. Transaction: The transaction is used to keep the consistency of data. Previous QoS model used the non-transaction factors to determine the QoS value. But, the potential users want to be provided with the procedure to rollback the service without any additional charge. The transaction can be evaluated in two ways. $q_{tx}(S)$ shows whether the rollback is supported and $q_{cons}(S)$ shows the time to rollback.

2.4.2. Compensation Rate: The compensation rate of Web service $q_{cons}(S)$ shows the compensation percentage of the existing performance price when the service provider fails to provide clear service or requested service.

2.4.3. Penalty Rate: The penalty rate of Web service is $q_{pen}(S)$ the penalty percentage of the existing performance price that has to be paid to the provider when the service requestor wants to cancel the service after the deadline to roll back the service.

3. QUALITY BROKER

The Quality Broker model has expanded the general Web service model for the users to have more efficient selection of Web services. The Quality Broker selects a service provider that matches the request to find the most optimized service for the Web application. Fig. 2 shows the determination levels of Web service execution using the Quality broker containing the user-oriented, non-functional factors.

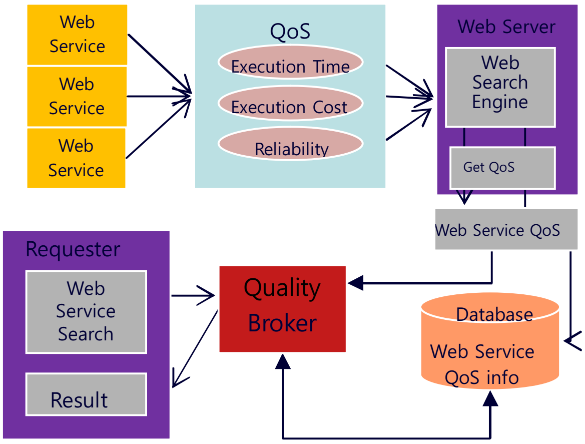


Fig. 2. The structure of the Quality Broker System proposed

3.1. Selection Process

The Quality Broker model has expanded the Web service model in order to efficiently select a Web service for the user. The Quality Broker selects the providers that match the request to select the most optimizing service for the Web application.

The application calls the Web service using the adaptor of Quality Broker to select an appropriate Web service for the request of Web application. The application itself does not contain the information of the Web service. The information of Web service is defined by SML data of the Quality Broker and the Quality Broker calls the provider's service by selecting an appropriate Web service from the Web service call of the application. The Quality Broker renews the QoS information in dependant on the result and records the information in XML data with the rank algorithm.

When implementing a Web application using the Quality Broker, WSDL information of Web service is entered into the Broker. The Broker creates DLL files with the information and records QoS information in XML data information after performing the initial QoS test. And the application developer enters the requested QoS to apply the rank algorithm inside the Quality Broker and record the rank of each Web service. When using a Web service in the application, the adaptor created in the Quality Broker is used instead of directly using the Web service. That is, it is a method to decrease the service dependency and complexity of development that the application can be changed according to the quality of Web service.

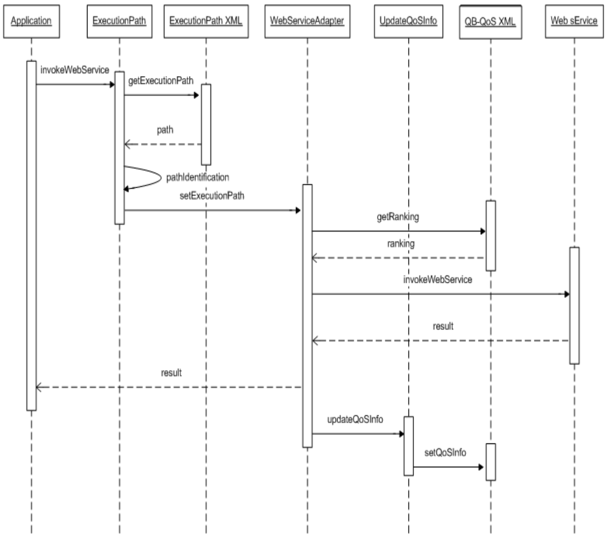


Fig. 3. Quality Broker Sequence Diagram

Fig. 3 shows the scenario to select a service in the Quality Broker. The Web application is active till it calls for a service and gets the result value. Likewise, the adaptor is active until it receives the result value. The application uses the features of Web service by using the adaptor. The non-functional aspect of service is fulfilled with the adaptor and the non-functional aspect is implemented through the Quality Broker and QoS SML. The specific QoS XML data will be described but the specific QoS factors considered in this study are performance time, performance cost, and reliability. Although the existing studies cover various QoS details, they have to be applied with complex mechanism and it is difficult to be implemented in Web applications because it considers unnecessary factors as well. This study defines the simple QoS XML data through XML Schema in this study and it selects the optimizing Web service in the Quality Broker with it. The ranking is defined internally and only interface is described for the user to easily use the service.

3.2. Web Service Determination in consideration of non-functional quality

QoS is specialized according to various factors of the Web service. For the QoS mode, we have studied the WS-QoS model and expandable model. There are 3 factors in determining the QoS model. The QoS factors are the execution time of Web service, execution cost of Web service, and reliability. Check the related factors in the execution level and perform the execution after determining the QoS factors and recording then in XML data. It is a method to satisfy the non-functional factor of Web service. The functional factor of Web service is determined in the level of execution determination through the Quality Broker. It is important to consider the functional and non-functional factors of Web service so that the Web service can be widely used in the business process because the Web service itself is software.

3.2.1. Service Execution Time : The execution time is a general factor to determine the performance of Web service. The execution time in applications using Web service is defined as the total time required for an instance from input to output. A web service with the fastest execution time possible is required in the application development under the time-based strategy. To measure the execution time, measure the time difference between the point to start a task and the point to move over to the next task. Such execution time is determined with the formulation below after measuring the time in the actual Web service call. The System Response Time Completed is the time that the response of system reaches to the user and the User Request Time is the time that the user sends the request.
Execution Time = System Execution Time Completed - User Request Time

3.2.2. Service Execution Cost : The execution cost is the cost related to the execution of Web service. It is necessary to evaluate the required cost for execution to secure the appropriate cost for the budget of firms and individuals for the application design and execution. The cost of Web service is the cost required when it is executed. There are 2 major factors in this cost: The cost for the Web service management and actual cost to run the service. The management cost is the cost related to the system monitoring and management. The actual cost of service is the cost related to the real-time service.

3.2.3. Service Reliability : The Web service reliability means the possibility of the service execution when the user runs the service. The reliability is divided into the system failure and process failure in consideration of the service failure. The system failure is the abnormal termination of the system that processes the service and software that runs the system, and the process failure is the exceptional process of service that causes an exceptional termination of the service. Each task has Early phase, Execution phase, and two clearly-defined termination phases which are the task failure and task success phases. Use the formulation below to calculate the reliability.
Reliability = 1 - (Number of failure/Number of execution)
Number of failures = Number of unsuccessful execution/Number of call for execution
Number of executions = (Number of execution failure or termination)/{(Number of failure or termination) + (Number of completion or success)}

3.3 Quality Broker XML Schema

It studies the structure of the QB-QoS factors and describes the QoS definition of Web Service based on QB-QoS XML Schema. QB-QoS uses XML Schema to define QoS like ExecutionPath.
Fig. 4 shows 4 factors in QB-QoS XML document.

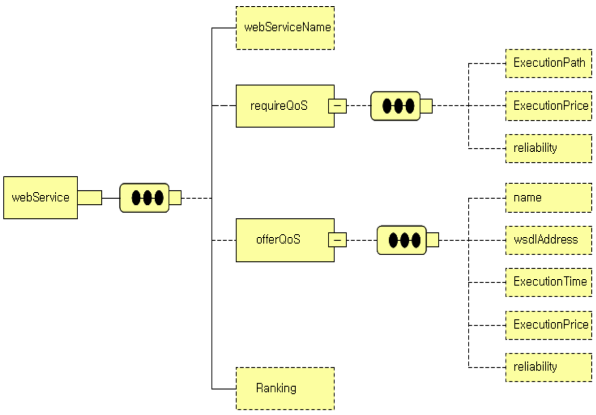
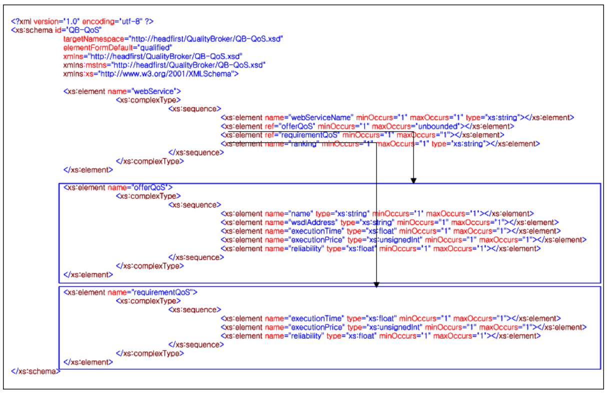


Fig. 4. The Structure of Quality Broker QoS Factor

webService is the most superior factor that defines QoS. webServiceName means the page that an application calls. requireQoS defines the QoS factor that the user request. User-defined QoS is divided into ExecutionTime, ExecutionPrice, and Reliability. requireQos includes only one in the webService definition. offerQoS includes the QoS definition of Web service updated by the initial QoS through the Quality Broker or execution.
The consisting factors are Web service name, wsdlAddress, ExecutionTime, ExecutionPrice and Reliability. offerQoS can have more than one definition in the webService definition. Tanking records Web service names in order of time, price, and reliability by comparing user-defined QoS and QoS of the provider.
Fig. 5 shows Quality Broker QoS XML Schema.



```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema id="QB-QoS"
          targetNamespace="http://headfirst/QualityBroker/QB-QoS.xsd"
          elementFormDefault="qualified"
          xmlns="http://headfirst/QualityBroker/QB-QoS.xsd"
          xmlns:mstns="http://headfirst/QualityBroker/QB-QoS.xsd"
          xmlns:xs="http://www.w3.org/2001/XMLSchema">

          <xs:element name="webService">
                    <xs:complexType>
                              <xs:sequence>
                                        <xs:element name="webServiceName" minOccurs="1" maxOccurs="1" type="xs:string"></xs:element>
                                        <xs:element ref="offerQoS" minOccurs="1" maxOccurs="unbounded"></xs:element>
                                        <xs:element ref="requirementQoS" minOccurs="1" maxOccurs="1"></xs:element>
                                        <xs:element name="ranking" minOccurs="1" maxOccurs="1" type="xs:string"></xs:element>
                              </xs:sequence>
                    </xs:complexType>
          </xs:element>

          <xs:element name="offerQoS">
                    <xs:complexType>
                              <xs:sequence>
                                        <xs:element name="name" type="xs:string" minOccurs="1" maxOccurs="1"></xs:element>
                                        <xs:element name="wsdlAddress" type="xs:string" minOccurs="1" maxOccurs="1"></xs:element>
                                        <xs:element name="executionTime" type="xs:float" minOccurs="1" maxOccurs="1"></xs:element>
                                        <xs:element name="executionPrice" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"></xs:element>
                                        <xs:element name="reliability" type="xs:float" minOccurs="1" maxOccurs="1"></xs:element>
                              </xs:sequence>
                    </xs:complexType>
          </xs:element>
          <xs:element name="requirementQoS">
                    <xs:complexType>
                              <xs:sequence>
                                        <xs:element name="executionTime" type="xs:float" minOccurs="1" maxOccurs="1"></xs:element>
                                        <xs:element name="executionPrice" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"></xs:element>
                                        <xs:element name="reliability" type="xs:float" minOccurs="1" maxOccurs="1"></xs:element>
                              </xs:sequence>
                    </xs:complexType>
          </xs:element>
</xs:schema>
```

Fig. 5. Quality Broker QoS XML Schema

3.4. Quality Broker Execution Level

Quality When implementing a Web application using the Quality Broker, enter the WSDL information of Web service in the Quality Broker. The Quality Broker creates a DLL file with the information and records the QoS information in the XML data information by performing an early QoS test. And the application developer enters the QoS required to record the rank of each Web service through the rank algorithm using the QoS information inside the Quality Broker. When using a Web service in the application, the adaptor created in the Quality Broker is used rather than directly using a Web service. It is for the application to reduce the service dependency and complexity of development that can change according to Web services.

3.5. Service Selection Process

The Web service selection process consists of the following as in Fig. 6: QoS Input, QoS Selection, QoS Evaluation, Decision Making, and Service Selection.

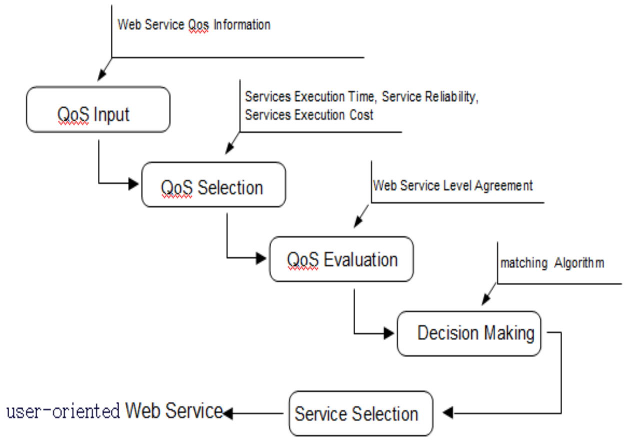

Web Service Qos Information
QoS Input
Services Execution Time, Service Reliability, Services Execution Cost
QoS Selection
Web Service Level Agreement
QoS Evaluation
matching Algorithm
Decision Making
user-oriented Web Service
Service Selection

Fig. 6. Service Selection Process

In QoS Input, all quality information related to Web service is entered. In QoS Selection, the properties in the quality properties entered in the previous step are selected. In QoS Evaluation, a service level must be defined for the selected quality properties and evaluated values are deduction according to the evaluation standard. Lastly, in Service Selection, the user-defined Web service is selected by comparing the Web service providers in the higher priority.

4. COMPARISON AND ANALYSIS

The existing Web service architecture uses the method to search a Web service from UDDI and is totally dependent on UDDI when implementing while the Web service broker uses UDDI through the Web service broker when implementing. The Web service broker immediately provides the information for a Web service when there is a Web service that the user wants but when there is no QoS information for the desired Web service, it searches the Web service through UDDI and tests the result of Web service. And then it delivers the information of the Web service to the user. The Web service broker is dependent on UDDI when executing. the Quality Broker is implemented using UDDI just like the Web service architecture.

The user can initialize the QoS factors of the Web service searched using UDDI and select a service using the QB-QoS XML data information of the Quality Broker when using the service.

However, the Web service architecture may have unsatisfactory cases when executing QoS of the Web service determined in implementation. It causes the service not to work and as a result, the application does not work. To complement the problems, the Web service broker uses UDDI when executing. But, if there is no information of Web service in the Web service broker, it induces the complexity to test QoS of all Web services and the increased execution cost as well as the increased execution time. This thesis suggests the Quality Broker to complement such problems..

<Table 1> is a table that compares and analyzes the Quality Broker considering the non-functional factors.

Table 1. Quality Broker Comparison considering non-functional factors

Standard System	Definition of QoS	Execution Level Change	UDDI Dependency	Complexity
WS-QoS Schema	User-oriented QoS definition available	Impossible to change in the execution level	Independent	Increased
Web Service Broker	User-oriented QoS definition unavailable	Impossible to change in the execution level	Dependent	Increased
QoS Registry	User-oriented QoS definition available	Impossible to change in the execution level	Dependent	Increased
Quality Broker	User-oriented QoS definition available	Possible to change in the execution level	Independent	Decreased

5. CONCLUSION AND FUTURE STUDIES

This paper has suggested the Quality Broker that can satisfy QoS of Web service and define the execution levels. It complements the Web service architecture and non-functional factors of Web service broker. The Web service user defines the early Web service QoS to use the Quality Broker which considers non-functional factors and is not dependent on UDDI. The Quality Broker has solved the problems that the existing Web services do not satisfy the non-functional factors by defining the user-oriented factors. With the suggestion of the Quality Broker, it brings out the effect to reduce the UDDI dependency and execution complexity in comparison to the existing Web service architecture and Web service broker. For future studies, it is necessary to add dynamic Web services that search the Web services required when practicing rather than the Web service search with UDDI and save them in the Quality Broker. Also, it is required to add a QoS determination method that the execution cost is added to the header of SOAP to be transmitted.

Systems in the future will be active using Web service that can be used and synchronized with other systems instead of being used in closed environment. Web service will be the main method of the future Web applications. The Web service definition in consideration of QoS and the method to automatically and dynamically insert Web service in the execution process are to be researched.

REFERENCES

[1] S. Ran, “A model for web services discovery with QoS”, *ACM SIGecom Exchanges, Vol.4, Issue.1*, Spring. 2003, pp1-10.

[2] Ovum, “Web Services Marker Overview”, *Ovum Research report*, Sept. 2002

[3] Nilo Mitra, "SOAP Version 1.2 Part 0: Primer", *W3C Recommendation*, 24. June. 2003.

[4] Julian Day, "Selection the Best Web Service" Proceedings of the 2003-2004 Grad Symposium, CS Dept, University of Saskatchewan, 7-8. April. 2004.

[5] Roberto Chinnici, Martin Gudgin Jean-Jacques Moreau, Jeffrey Schlimmer, Sanjiva Weerawarana "Web Services Description Language(WSDL) Version 2.0 Part 1:Core Language", *W3C Working Draft*, 3. August. 2004.

[6] http://www.uddi.org

[7] M. Tian, A. Cramm, H. Ritter, and J. Schiler, "Efficient Selection and Monitoring of QoS-aware Web Services with the WS-QoS Framework", *The 2004 IEEE/WIC/ACM International conference on Web Intelligence*

[8] M. Tian, A. Gramm, T. Naumowicz, H. Ritter, J. Schiller, "A Concept for QoS Integration in Web Service", *Web Information Systems Engineering Workshops* 2003, *Proceedings Fourth International Conference on*, 13. Dec. 2003, pp149-155.

[9] Y. Liu, A H. H. Ngu, Texa State University, L. Zeng, IBM T. J. Wastson Research Center, "QoS Computation and Policing in Dynamic Web Service Selection", *The Thirteenth International World Wide Web Conference*, May. 2004, pp66, 17-22.

[10] “Web Service Architecture" *W3C Working Group Note*, 11. February. 2004.

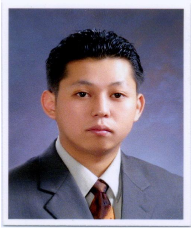
[11] J. O’Sullivan, D. Edmond, and A. ter Hofstede. What’s in a Service? Distributed and Parallel Databases, 12(2–3):117–133, September 2002.

[12] T.Bellwood, et al., “UDDI Version 3.0 Published Specification”, July, 2002.(http://uddi.org/pubs/uddi_v3.htm)

[13] S. Rogers, “Web Services Software 2004-2008 forecast”, *IDC*, Apr. 2004.

[14] C.Patel, K. Supekar and Y.Lee, “A QoS Oriented Framework for adaptive Management of Web Service based Workflows”, *LNCS, VOl.2736*, Sept. 2003, pp826-835.

[15] L. Aversano, G. Canfora and A. Ciampi, “An algorithm for Web Service discobery through their composition”, *Proceedings of the IEEE International Conference on Web Services*, 2004.



Sung-Ho Sim
He received M.S. in Computer engineering from Kyunghee University, Korea in 2005.
Since 2005, he is taking the Ph. D. course in Computer engineering from Kyunghee University, Korea. His research interests include Web Service, CBSE, and Software recycling.

Young-Jae Song
He received B.S. in Electronic engineering from Inha University, Korea in 1976. He also received M.S. in Computer engineering from Keiko University, Japan in 1976. In 1979, respectively, he received Ph.D. in Computer Science from Myoungji University, Korea.
He joined as a researcher in Seiko, Japan from 1971 to 1973. During 1982 to 1983, he served as research professor in Maryland University, U.S.A.
From 1976 till present he serves as a professor in College of Electronics and Information at Kyunghee University, Korea.
His research interest includes Web Service, CBSE, CASE Tool and Software Recycling.