WEB SERVICE RETRIEVAL AND SELECTION WITH SMART APPLICATIONS: A QUALITY APPROACH

A dissertation submitted
to Kent State University in partial fulfillment of the requirements for the degree of Doctor of Philosophy

by
Amany K. Alnahdi

May, 2017
Dissertation written by

Amany K. Alnahdi

M.S., California State University-Fresno, USA, 2011
B.S., King Abdulaziz University, Saudi Arabia, 2004

Approved by

Austin Melton, Chair, Doctoral Dissertation Committee
Mikhail Nesterenko, Members, Doctoral Dissertation Committee
Angela Guercio
Alan Brandyberr
Oana Mocioalca

Accepted by

Javed Khan, Chair, Department of Computer Science
James Blank, Dean, College of Arts and Sciences
# TABLE OF CONTENTS

LIST OF FIGURES ............................................................................................................... VII

LIST OF TABLES ................................................................................................................ IX

ACKNOWLEDGEMENTS ...................................................................................................... X

CHAPTER 1 INTRODUCTION .............................................................................................. 1

1.1 Problem Description and Motivation ................................................................. 1

1.2 Quality Issues to be Addressed in Web Service Retrieval Domain ............... 3

1.3 Research Contribution ...................................................................................... 3

1.4 Research Organization ...................................................................................... 5

CHAPTER 2 LITERATURE REVIEW ................................................................................. 6

2.1 Information Retrieval ....................................................................................... 6

2.2 Quality Function Deployment and its Uses in the Software Domain .......... 8

2.3 Web services and Quality Web Service Retrieval ........................................ 11

2.4 Web Service Retrieval and Semantics .......................................................... 13

2.5 Measurement .................................................................................................. 17

2.6 Word Net Semantic Similarity Applications ............................................... 18

CHAPTER 3 DISSERTATION ROAD MAP .................................................................... 21

3.1 Definitions ...................................................................................................... 21

3.2 Dissertation Roadmap .................................................................................. 23

3.3 Summary ....................................................................................................... 23

CHAPTER 4 WEB SERVICE RETRIEVAL BASED ON QUALITY ............................... 24

4.1 Semantic Web Service Retrieval Model ...................................................... 26
6.1 Steps of Systematic Analytical Approaches for Identifying Quality Service Requirements of Web Service Artifact

6.1.1 Collecting Web Service Quality Requirements ................................................. 64
6.1.2 Filtering Web Service Quality Requirements .................................................. 65
6.1.3 Prioritizing Web Service Quality Requirements .............................................. 66
6.1.4 Classifying Web Service Quality Requirements .............................................. 66

6.2 Model Evaluation ........................................................................................................ 67
6.2.1 Experiment Settings ............................................................................................. 67
6.2.2 Experiment Results .............................................................................................. 69

6.3 Fully Automating the Steps of WSDQFD ................................................................. 70

6.4 Summary ................................................................................................................... 71

CHAPTER 7 .......................................................................................................................... 72

USING QUALITY FUNCTION DEPLOYMENT (QFD) TO DESIGN A SMART SCHOOL QUALITY FACTOR MODEL: INTEGRATING QFD INTO IOT .............................................................................................................. 72

7.1 State of art of QFD and IoT ....................................................................................... 74

7.2 The process for Building a Quality Factor Model for A smart school ................. 77
7.2.1 Deciding on the Stakeholders for Smart Schools Using IoT ............................ 78
7.2.2 Determining Quality Factors from Stakeholders ............................................. 78
7.2.3 Filtering and Prioritizing Quality Factors ....................................................... 82
7.2.4 Building a Quality Factor Model ..................................................................... 83
7.2.5 Normalizing Quality Factor Model .................................................................. 85
LIST OF FIGURES

Figure 1. Mapping Hubbard measurement components to Web service quality similarity ................................................................................................................................. 25

Figure 2. An overview of the QoS semantic Web service retrieval model.......................... 29

Figure 3. QoS Ontology hierarchy using OWL Vis Tab in Portege ..................................... 31

Figure 4. Extension process of OWL-S TC4 ........................................................................ 32

Figure 5. Extension of ServiceCategory and ServiceParameter that includes quantitative values for QoS parameters.......................................................................................... 33

Figure 6. QoS retrieval based on minkowski distance .......................................................... 35

Figure 7. Property reasoning .................................................................................................. 37

Figure 8. Deterministic approach Web service retrieval ..................................................... 38

Figure 9. Adaptive approach Web service retrieval ............................................................. 39

Figure 10. Precision and recall ............................................................................................... 44

Figure 11. Applying normalization to a dataset ................................................................... 50

Figure 12. Web Service quality dataset distributions ............................................................. 51

Figure 13. Web service retrieval based on quality ................................................................ 53

Figure 14. Web service ranked results ................................................................................ 54

Figure 15. QWS1 Clustering visualization result using weka .............................................. 56

Figure 17. Clustering assignment for QWS file in Weka QWS ........................................... 57

Figure 16. Clustering model using MakeBasedDensity clusterer .......................................... 57

Figure 18. Measurement components of applying QFD to Web service description .......... 63
Figure 19. Steps of systematic analytical approach for identifying quality of service requirements of Web service artifact ................................................................. 64

Figure 20. WSDQFD Evaluation Survey Questions ................................................................. 68

Figure 21. Process for building a quality factor model for smart schools ...................... 78

Figure 22. Quality matrix for smart school Web services ..................................................... 84
LIST OF TABLES

Table 1. Quantified representation of QoS parameters.......................................................... 34
Table 2. QWS dataset description.......................................................................................... 47
Table 3. Precision and recall for original QWS1 dataset....................................................... 58
Table 4. Precision and recall for normalized QWS1 dataset ................................................. 58
ACKNOWLEDGEMENTS

I would like to thank everyone who has helped me during the journey of my study. I would like to thank my professors Dr. Austin Melton and Dr. Alex Liu for their academic support through my graduate study. I also would like to thank my family and friends for their support.

Amany Alnahdi

April, 2017, Kent, Ohio
CHAPTER 1

Introduction

Web services are software module components that provide specific functionality over the Internet. When several Web services provide the same functionality, quality plays an essential role in distinguishing between those services. Since different users have different quality needs, a quality query given by one requester is different than that of another requester. Retrieving a Web service that best matches the requester’s quality information is essential when differentiating between functionally equivalent Web services.

1.1 Problem Description and Motivation

Programming using Web services is a software methodology trend that provides several benefits. The authors of (Graham et al., 2001) fully described the advantages of using Web services as a software platform, including platform independence, implementation hiding, and the ability to provide loosely coupled applications. In addition, the authors emphasize the benefit of using Web services as a software platform when it comes to the business domain, so that the businesses can concentrate on their own business instead of spending most of their efforts on the implementation of the software.

Quality of Service (QoS) for a Web service is described by non-functional attributes of a Web service (Xianglan, Yangguang, Bin, & Gang, 2011). The task of
prioritizing Web services based on quality involves important factors. One of the challenges of doing research on Web service quality is that data on Web services is limited. Also, systems that retrieve Web services based on QoS have limited availability. This dissertation gives an infrastructure model for retrieving Web services based on quality. This research investigates quantifying the similarity of a Web service query to available Web service offers with respect to quality.

A quality perspective for one user is different from that of another user. In other words, what represents desirable quality for one user might be undesirable for another user. For example, let us say that user A is looking for a printing service for a draft for an essay. Since user A would not worry about text appearance, so low printing resolution is the optimal quality for him. On the other hand, if User B is looking for a printing service for a final essay, high printing resolution represents the right quality for him. A Web service that provides printing functionality may have quality parameters such as \( QP=\{qp_1, qp_2, qp_3, \ldots, qp_n\} \), for example, speed, accuracy, resolution, etc. A Web service requester WR1 who is looking for an online printing Web service to send business documents to a company might search an online registry for Web services with some set of quality values \( WR1QP=\{v_1, v_2, v_3, \ldots, v_n\} \). Another Web service requester WR2 might have another set of quality values \( WR2QP=\{v_1, v_2, v_3, \ldots, v_n\} \). The two requesters might have different quality values for different quality parameters such as \( WRQ1= \{80\%, 20\%, 30\%, \ldots, 45\% \} \) and \( WRQ2=\{50\%, 10\%, 60\%, \ldots, 70\% \} \). The Web service search pool might have several functionally equivalent Web services (printing in this case). So, the requester might need to further filter those Web services
based on quality. The above example describes how quality differs with the perspectives of different users of Web services for programming and business services. Web services also can be of a purely software nature.

Considering quality retrieval for purely software Web services is also important. An example of a purely software Web service is a data analysis Web service that performs calculations on statistical information about a given set of data such as average, standard deviation, and visualized charts. Quality characteristics for such a Web service would include speed, accuracy, scalability and price.

1.2 Quality Issues to be Addressed in Web Service Retrieval Domain

- How Web service descriptions can be improved so they can support Web service quality retrieval?
- Evaluating Web service selecting strategies based on quality.
- What are the procedural methodological steps that ensure the quality of a Web service description?
- How can these steps of maintaining the quality of a Web service be automated?
- How can Web service quality be applied to the domain of the smart schools and the Internet of Things (IoT)?

1.3 Research Contribution

The contributions of this dissertation to Semantic Web research have several folds. First, enhancing the Web service description improves and consolidates quality
attributes and values. For this purpose, a QoS ontology built based on specifications according of the World Wide Web Consortium (Lee, Jeon, Lee, Jeong, & Park, 2003), a QoS extension model to the OWL-S_TC (M. Klusch, Kapahnke, Fries, Alam Khalid, & Vasileski, 2010), experiments of similarity measurement and parameter control algorithms for service retrieving, and data mining techniques as a means to compute relevance sets.

Second, this dissertation proposes extending the Web service quality description to involve quality characteristics that would enable quality based Web service retrieval. In addition, in this dissertation provides quality retrieval algorithms and applied them on both simulated data and a real dataset provided by (Al-Masri & Mahmoud, 2007a) (Al-Masri & Mahmoud, 2007b) (Al-Masri & Mahmoud, 2008a) by elaborating our work in literature (Alnahdi, Liu, & Melton, 2015). The importance of the research is to measure the accuracy of a Web service retrieval that would help Web service requesters and providers in making a decision when differentiating and ranking between different Web service offer choices.

Moreover, this research discusses applying the concept of Quality Function Deployment (QFD) from the industry domain to the domain of Web services. Specifically, this research focuses on Web service description artifacts that describe the Web service quality information. Finally, this dissertation contains a chapter which is dedicated on identifying quality parameters of Web services for building smart schools.
1.4 Research Organization

This dissertation is organized as follows: Chapter 2 explores the research work that has been done in Quality Function Deployment (QFD) and House of Quality (HoQ), Web service, and the semantic Web; Chapter 3 gives an overview of the research methodology and research components of this dissertation; Chapter 4 gives detailed information about the methodology of Web service selection based on quality; Chapter 5 provides detailed information about information retrieval for Web services based on quality when applied to a real dataset; Chapter 6 provides our perspective on Quality Function Deployment (QFD) and Web service descriptions; Chapter 7 presents information about identifying quality factors for smart schools that use the IoT; Chapter 8 provides the conclusion of our work; and Chapter 9 sets out the future work of this dissertation.
CHAPTER 2

Literature Review

This dissertation approached the problem of selecting Web services based on quality. Web service selection is an information retrieval problem in which a user who is looking for a Web service has some functional and quality specifications. Once the functional specifications are met, quality plays an essential role in filtering and ranking those Web services. This section is designated for reviewing the literature on the domain quality and Web service quality retrieval. The literature review of this section is organized and classified based on subcategories of related work.

2.1 Information Retrieval

Information retrieval is an essential task for computer users. Because computers and the Internet has enabled people and companies to store huge amount of data in databases either in personal computers or on external servers connected to the Internet. The stored data will be useful when users are able to retrieve them and benefit from the information. Web service retrieval is an information retrieval problem. In computer science literature, there have been several books and research papers that approached the domain of information retrieval. As follows, this section reviews one of the main information retrieval resource and how it is related to the area of this dissertation.

The author of (Korfhage, 1997) emphasized in his book that people’s need for information is frequent and essential to solve problems and make decisions ranging from
sophisticated to naïve, for example, medical diagnoses to crossing streets. The author listed various information resources, such as libraries, experts, computers, and friends. As the author specified, computers, such as personal computers, carry huge amounts of data that hold many gigabytes worthy of memory. In addition the author presented algorithms for enhancing information systems by improving their information retrieval functionalities.

The author of (Korfhage, 1997) indicated that designing an efficient information system is the work of a specialist. Also, the book discussed essential topics such as information storage and information retrieval. As the author specified, information storage is the base phase for the information retrieval task. Also, he specified that such information has to be stored in a clear and organized form so it can be retrieved efficiently. In addition, the author mentioned that computer based information systems need information to be digitized so it can be stored. Also, the author categorized two main components of an information system; they are the ectosystem and endosystem. An ectosystem includes variables that are beyond the designer’s control. On the other hand, an endosystem has factors that designers can decide.

This dissertation envisions Web services as a software paradigm for building software systems. The adoption of Web services from different software companies has to lead to producing a plethora of Web services. It is essential for the Web service user to have ability to access and retrieve available Web services provided in the Internet. Having methods for Web service retrieval enable Web service users to make decisions on selecting Web services that fit their needs.
2.2 Quality Function Deployment and its Uses in the Software Domain

When one has multiple service options, there is a necessity to rank those choices based on quality. Quality is an important aspect in any organization when making decisions. The literature has approached quality in several domains. The following sections provide information about how quality has been addressed in variety of domains including industrial, computer science, and software.

(Chan & Wu, 2002) presented a survey to review the literature in the quality function deployment (QFD) domain. Their survey paper covered and classified 650 QFD publications via searching several databases. Their resourceful survey is basically a good reference for researchers who are interested in learning about QFD or using QFD in their domain. The authors mentioned that QFD was initially developed in Japan in the late 1960’s, and it spread gradually in the 1980’s in the U.S. And then other nations such as Australia, Belgium, Brazil, Denmark, and Germany have adopted this concept. The authors gave historical information about the QFD and how it was first initiated in Japan and the U.S. They also provided a categorization analysis of QFD applications in terms of the reasons (why) and what industries (where) those applications have been applied. In their paper, the authors pointed out useful resources for QFD: Organizations which offer training and consulting in QFD. Other resources, such as software solutions, that facilitate the QFD process, including design and evaluation functionality; online information that is available to users, including websites, email discussion lists, and forums; and QFD literature such as books, articles, journals, and conference proceedings. Then, the authors listed essential functions of QFD, such as product development, quality
management, customer-needs analysis, design planning, decision making, engineering, timing, management, and cost. Although, the authors listed some specific fields of applications, they also claimed that QFD can be applied to any other domain and there are no limitations in applying QFD applications. The authors in their research elaborated on the explanation of two well-known applications of QFD namely product development and quality management. However, they claimed that the QFD can be applied to other domains.

As can be seen from the authors’ of (Chan & Wu, 2002) perspective, the use of QFD can be applied to other domains. In this dissertation, applying the concept of the quality can be explored for many perspectives including quality of Web service selection, producing a quality Web service artifact, and quality Web service smart platform applications.

In his research (Hauser, 1993) discussed how the House of Quality (HoQ) concept was applied in real business applications. Hauser said that business owners are not interested to hear the frequently given advice, “the best way to design a good product is to listen to customer.” Instead, they are interested in following a step by step procedure to design a quality product. Hauser said that some HoQ applications reduced design times by 40% and design costs by 60% while maintaining, or even improving the quality of the design. As the author discussed, the basic idea behind the quality can be achieved by hearing the “Voice of the Customer” and the “Voice of the Engineer.” The steps of hearing the voice of the customer include identifying customer needs, prioritizing the needs, and comparing customer perceptions. The phases of building HoQ include
identifying design attributes, comparing engineering measures, developing the relationship matrix, developing the roof matrix, and making other estimates.

As can be seen from the author of (Hauser, 1993), the HoQ concept has proven successful, since it has reduced time and cost in the business domain. Also, the author indicated that listening to the voice of customers and engineers is essential in the process of applying the HoQ. Different from what the authors mentioned, this dissertation identifies more stakeholders when applying the concept of HoQ to the domain of Web services, including Web service users, Web service engineers, Web service providers, and research engine holders.

The author in his research (Liu, 2000) mentioned the improvement of software quality is essential in designing quality software and that it would lead to reducing updates in the software development life-cycle, including requirements such as collection, design, programming, and maintenance. As the author mentioned, the first country that used Software Quality Function Deployment (SQFD) is Japan. Then, well-known companies, such as IBM and Hewlett-Packard, began to use it in other countries. Software applications that applied SQFD include a management information system, decision support system, and operating systems. Also, the author stated that QFD was used for both creating new software systems, and upgrading available systems.

A web service is a software methodological trend that is used to build big systems. This dissertation highlights on applying quality to the domain of Web services, thus facilitating the process of retrieving and building Web services. Moreover, this
dissertation emphasizes that putting more effort on building a quality Web service interface artifact is essential when it comes to Web service retrieval.

2.3 Web services and Quality Web Service Retrieval

Web services enable users to get software services over the Internet. When multiple Web services provide the same functionality, a user might differentiate and make a selection between those services based on quality. The following section reviews some of the quality Web service selection literature work.

The authors of (Xianglan et al., 2011) presented a survey on QoS-Aware Dynamic Web Service Selection. They distinguished between two aspects of choice: the main body making the selection (a person or several people) and the basis of the selection. They also claimed that there is a lack of supporting multi-user services, selection, and evaluation. Also, the authors classified service selections into two aspects: component service selection and composition service selection. The difference between those two categories of selection depends on how many Web services are retrieved during the selection process. The component service selection is when the user has to retrieve one Web service. Composite service selection occurs when several Web services are retrieved and must specify a QoS overall expectation.

This dissertation approaches component service selection by providing a numerical methodology based on distance calculation between a query and offers. Also, this dissertation provides a vision of the HoQ concept which involves several stakeholders in the Web service domain who have their own specifications when it comes to Web service quality applications.
The authors of (Moo-Mena et al., 2009) proposed a statistical model to diagnose the behavior and the state of a Web service application. The authors classified the state of an application into three categories: normal, degraded, and broken. They represented their data distribution based on a BoxPlot method. The authors also proposed a three phase way for supporting a Web service architecture, specified as monitoring, diagnosis, and recovery. Their diagnosing modules follow several steps: First, they read QoS values from a database, and then they monitor large volumes of data. Once it arrives, the modules analyze the data, and then evaluate the degraded scenario. The authors performed a test case on a digital library application. The authors approach the issue of maintaining Web service applications by monitoring the behavior of this application. This dissertation approaches the issue of maintaining the availability of Web services by pre designing a quality Web service interface artifact.

The authors of (Al-Masri & Mahmoud, 2007a) said that Web services provided to deploy applications are to be used for business-to-business integration. The authors specified challenges which hinder the adoption of Web service technology. These challenges include defiance of using approaches of computing and monitoring quality characteristics in a fair and transparent way, along with the limitation of including quality parameters in the discovery process. The authors created a Web Service Repository Builder (WSRB) architecture and extended that repository by offering QoS support. The authors mentioned that their solution has a high success rate in providing the most relevant Web services. They also created a QoS Manager (WS-QoSMan), a service broker, to create an intelligent system that examines Web services’ quality criteria in a
transparent way. In addition, the authors introduced a metric called Web Service Relevancy Function (WsRF). It measures relevancy and then ranks Web services based on both user preference and QoS metrics. In this dissertation, the WsRF metrics was used in the proposed similarity numerical model as part of QoS attributes.

2.4 Web Service Retrieval and Semantics

In general, semantics are used to explain the meanings of terms. In Web pages, semantics are used to enrich the html text so it can involve more information to be exchanged, and processed by computers. In the Web service domain, the use of semantics has been approached by using ontologies and semantic rule languages. This section provides several research papers that involved semantics in Web services.

In (Lu, Hsu, & Kuo, 2013), there is a proposed matchmaking approach that combines WordNet, domain ontologies, and Semantic Web Rule Language (SWRL) rules. These three components are used by the authors as follows: WordNet is used for query extension; ontologies are used for storing service information such as service name, and SWRL rules are used for enhancing relationships among concepts in ontologies. According to their results, this approach provided a higher hit rate than keyword methods. In addition, the use of WordNet provides the requesters with more synonyms. This dissertation used semantics in the domain of the quality where we have proposed incorporating quality attributes in the interface of Web service description. Also, this research proposed an ontology to describe Web service quality attributes and the reason behind the relationships between the quality ontology concept using semantic Web rules.
The authors of (Pedersen, Patwardhan, & Michelizzi, 2004) provided a free software package called “WordNet::Similarity” to compute similarity and relatedness between concepts. Their software provided six similarity measures and three relatedness measures. The infrastructure of their measure is the WordNet database. The input to their program is two concepts whose similarity is to be measured and the calculated output is a number that represent the relatedness degree. Similarity measures are used to compute relations such as an “is-a for hierarchy;” for example, a penguin is a bird, and A is like B, for example a laptop is like a notebook. The authors used the WordNet hierarchy because it is a very suitable way to organize verbs and nouns into a hierarchy. On the other hand, there are other relations that connect concepts such as part-of, is-opposite-of, is-made-of, is-an-attribute-of. The six similarity measures provided by “WordNet::Similarity” have three of which are based on calculating information content of the least common subsumer (LCS), the highly specific concept is shared as an ancestor by a pair of concepts. The other three similarity measures between two concepts are based on path lengths. On the other hand, relatedness measures are more general; for example, HSO metric in the software package is used to classify direction relationships between two words by finding a path. Other measurement techniques, such as Lesk and vector use information from WordNet glosses, definitions for a concept. Lesk is based on finding overlaps between two concepts of glosses X and Y, while the vector relatedness measure is based on the co-occurrence matrix. The interfaces of WordNet::Similarity include command lines and Web-interfaces. WordNet::Similarity can be plugged inside a Perl program by calling methods, or it can be downloaded from the website.
This dissertation mostly focused on quality attributes of a Web service description; typically, these quality attributes have numerical values. However, a suggestion using these similarity measures to further describe the quality of string attributes was introduced.

The authors (Antoniou, Groth, Harmelen, & Hoekstra, 2012) pointed out in their book that the goal of the semantic Web domain is to “make the web more accessible to computers.” The old versions of the Web display text and pictures. This type of information has to be processed intelligently by human beings for selecting and combining information. The semantic web makes the Web more knowledgeable for machines. Also, it provides search possibilities using keywords and meanings of those keywords, i.e., the semantics. The authors have classified some basic technologies that can be used in the semantic Web such as labeled graphs, Web identifiers (Uniform resource Identifiers – URI), and ontologies.

The authors of (Antoniou, Groth, Harmelen, & Hoekstra, 2012) originate the term ontology with the philosophy domain that describes the meanings of concepts; the same concept can have several meanings in different domains. To illustrate the authors’ idea, we provide an example of a term that has several meanings in different domains. The term tree in the furniture domain represents a tall, thin wooden object used to hang clothes and hats on. In the computer science domain, and specifically in the algorithms field, tree is a term used to represent the structure that holds data in multiple levels and enables search algorithms such as a Breadth First Search (BFS). In another domain, such as nature, tree represents a type of plant, e.g., an apple tree. Another meaning for tree, in
genealogy, is the structure that represents the relationships between family members or tribes. Similarly, python has different meanings in different contexts. In animal science, it means a type of snake. However, in computer science it is a name of a programming language. In addition, in their book the authors demonstrated that the search process can be enhanced by not only looking to keywords but also observing meanings of those words and the context of the search phrases.

The book (Antoniou et al., 2012) also indicated some of the design principles that made the Web more semantic by enabling structured and semi-structured data to the Web. As the authors mentioned, there are three basic technologies for the semantic Web, labeled graphs, Web identifiers, and ontologies. Labeled graphs are used to describe objects and relationships between objects. Web identifiers (URI) are the identification units for Internet resources. Ontologies are data structures that hold vocabularies of a specific domain of knowledge in a hierarchal pattern. The authors also clarified that the traditional Web has a distributed nature, which they describe according to location and ownership. They explained that different Web pages are linked together. Moreover, the book distinguishes between programming languages designated for the Web, such as HTML and XML. HTML is the leading Web programming language, largely because it can be written by different tools to present information to people. This language is designed to present information in a readable format to people. However, the deficiency of this language is that it is not appropriate to be understood by programs. On the other hand, XML is used to define the structure of the Web page. While the book emphasizes the benefit of using semantic Web in Web pages, this dissertation emphasizes the use of
semantic Web in the domain of Web services where semantic features are incorporated in the Web service interface so users can benefit while retrieving the Web services.

In his book (Yu, 2011) the author provided a practical guide for the semantic web. The author indicated that since 2001, the semantic Web provided chances for both academic research and real applications. The book gave both directions How-To and What-Is for Semantic Web domains.

This dissertation provides applicable methodology for the use of semantics in the domain of Web service and incorporates semantics to the Web service description artifact.

2.5 Measurement

To evaluate the performance of a certain task, a measurement procedure has to be done. When it comes to the Web service retrieval domain, we would like to evaluate algorithms to be used in the retrieval process. The following section explores what the literature has indicated about the necessity of measurement.

In the book (Hubbard, 2007), the author claimed that things that are commonly assumed not to be measureable can be determined, including intangible concepts such as budget, flexibility, risk, and quality. As the author specified, measuring quantifies things to enable decision makers to make cognizant decisions which are based on information and measurement. This process reduces both the error rate and resource misallocation while it decreases the possibility for rejection of good ideas. Essentially, the author assumed that anything which can be observed can also be measured. The author illustrated that the foundation of the measurement field is based on the information theory.
field proposed by Claude Shannon in 1940s. Hubbard illustrated that Shannon’s book, “A Mathematical Theory of Communication,” led to the establishment of the domain of measurement and information theory. In addition, Hubbard stated that the basic idea of information theory is to “reduce uncertainty.” The author also illustrated that the measurement field is used to map an object to be measured to a number. By specifying the three measurement components (the concept to be measured, the object, and the methodology of measurement), a measurement problem can be well defined. The author also showed that the measurement field is related to the statistics field, based on statistician David Moore’s phrase, “If you don’t know what to measure, measure anyway. You’ll learn what to measure.” Moreover, Hubbard emphasized the benefit of calculating the value of information. Also, he provided methodologies of information analysis that can be accessed from the measurement website (How To Measure Anything, 2014). Just as the significance of measurement is essential and emphasized by the author, this dissertation also emphasizes the significance of measuring the similarity of a Web service query to Web service offers to enable Web service users to make informed decisions when differentiating between several Web services that have similar functionality, and specifically when users make a selection based on Web service quality.

2.6 Word Net Semantic Similarity Applications

The semantic Web and Web services are interdisciplinary fields. There are some researchers that have worked in combining the semantic Web and Web services. This section explores some of the literature integrated into the semantic Web.
(Chen, Jianzhuo, Liying, & Bin, 2009) proposed two similarity algorithms to measure the semantic similarity between two words: the Hierarchy-Spread (HS) algorithm and the Bi-Direction One Step (BDOS) algorithm. The HS algorithm is mainly based on the hyponym and hypernym of the words when measuring the similarity. On the other hand, BDOS is based on measuring the relation starting from two, and the BDOS deals with up to four relations. The authors define semantic distance to be the path length between two words in the WordNet directed graph. The HS algorithm works as follows: it expands a word into a hyponym/hypernym and puts it in a finite set called Hesets. Word sets (WS1 and WS2) contain the synonym of the words to be compared and find the common parent node. Once the parent node is located, there will be a path to be measured between the two words W1 and W2, and they calculate the shortest path. The authors also proposed the BDOS algorithms that improve the limitations of the HS algorithm, mostly by enhancing the accuracy. They provided a dynamic threshold in this algorithm for providing control. The authors have compared their algorithm to Yang’s algorithm. The results reported that Yang’s algorithm provides less accuracy than the HS and BDOS algorithm’s. Yang’s algorithm’s resultant distance is longer than the other two when running the program on 80 lexical pairs of words which demonstrates that BDOS and HS consume similar time and BDOS provides higher accuracy while keeping longer time complexity. BDOS also causes an expansion of nodes due to its nature of dealing with all kind of relations.

The authors of (Ahsae, Naghibzadeh, & Yasrebi, 2010) mentioned that WordNet is an important source that is used for knowledge retrieval and natural language
processing. The similarity between two words can be measured by calculating the depth of the two words by subsuming and measuring the shortest path that connects them. The authors have improved semantic similarity by enhancing transfer functions of the old research. They also performed parameter tuning using swarm optimization and proved the improvement of the correlation results based on human judgment benchmarking.

The authors of (Keppeler, Brune, & Gewald, 2014) have introduced a semantic retrieval model that involves commercial attributes. Their semantic model focuses on semantic and commercial facts. They have proposed an XML-based description language tied to Web Service Description Language (WSDL). The purpose of their work is to involve semantic and commercial aspects when retrieving Web services. Since a service repository is not restricted to use a WSDL description, the authors proposed a language that does not directly extend WSDL. They claimed that generic schemes would provide more adaptability and flexibility. The structure of their language is composed of QoS attributes and service functionality attributes. This dissertation is different in that it enriches WSDL so that it supports QoS attributes.

The authors (Upadhyaya, Zou, Keivanloo, & Ng, 2015) introduced a method for mining QoS reviews from different domains. Their method processed a natural language and extracted Quality of Experience (QoE) attributes from user text reviews. Then, they represented these attributes in an automatic way. They concluded that most QoS and QoE attributes are correlated.
CHAPTER 3

Dissertation Road Map

This chapter contains the basic terms used in this dissertation, while also providing its main components, and describing the overall dissertation roadmap.

3.1 Definitions

Throughout this dissertation, we discuss some concepts and terminologies. In this section, we define these concepts and terms so their meaning is clearly determined.

- **Web Service** is modular software that provides certain software or service functionality over the Internet.

- **Web Service Description Artifact** is a software document that consists of tags and descriptions of a Web service in terms of functional specifications and quality specifications.

- **Stakeholders for Identifying Quality Factors of Web Services** are people or organizations who are involved in identifying quality factors to be included in the Web service description artifact.

- **Web Service Quality Function Deployment** is the process of deploying quality procedures in the defining of Web service description artifacts for the purpose of designing a standardized Web service description that enables fast retrieval of Web services.

- **Web Service Functionality** is a criterion that describes the main functional purpose of a Web service.
• **Web Service Quality** is criteria that describe certain non-functional aspects of a Web service.

• **Domain Independent Web Service Quality Characteristic** is a quality characteristic that describes general criteria of a Web service. By “general criteria,” we mean quality criteria that can be common to several Web services regardless of their functionality, for example reliability.

• **Domain Specific Web Service Quality Characteristic** is a quality characteristic that is related to functionality of the Web service; for example, resolution for printing a Web service. This quality characteristic is for Web services in the printing domain.

• **Quality of Service Attribute** is a software feature that describes common characteristic of a Web service for example *transmission time*.

• **Quality of Service Type** is the type of the quality of service value such as numerical, categorical, or string.

• **Quality of Service Value** is the actual value associated to a specific Web service quality.

• **Web Service Composition** is process of combining multiple Web services that performs specific functionalities to generate a Web service of combined functionality.

• **A Quality Based Web Service Information Retrieval System (QBWSIRS)** is a system that involves different components to enable storing, searching, and measuring qualities of a Web service.
• **User of QBWSIRS** is an individual or a software company that uses the QBWSIRS for searching Web services based on quality.

• **Provider of QBWSIRS** is an individual, a software company, or a party that provides the QBWSIRS for searching Web services based on quality.

3.2 **Dissertation Roadmap**

This dissertation addresses several issues in the domain of Web services and quality. Specifically, this dissertation starts by working on a quality based Web service information retrieval system. The proposed model provides a selection strategy that provides a Web service requester a mean, to inquire about a Web service based on quality characteristics. Then we investigate the applicability of using Quality Function Deployment (QFD) in the domain of Web service retrieval, specifically on a Web service description artifact. Then, we explore identifying quality factors in the domain of Web service specifically in the application of smart schools that use the Internet of things (IoT).

3.3 **Summary**

This chapter defines some of the main terminologies that are used throughout this dissertation. It also provides brief road map on how this dissertation is organized.
CHAPTER 4
Web Service Retrieval Based on Quality

This research measures the similarity between a Web service query and Web service offers based on quality. Measuring quality characteristics is an essential component when building an information retrieval system based on quality. To measure the similarity, algorithms for choosing a good provider’s service were applied. To test our methodology, we applied those algorithms on two settings: simulated quality data and quality dataset that describes quality characteristics of real Web services. In this chapter we present the similarity algorithms applied on the simulated quality data. Then the real dataset is described and the retrieval model is presented in Chapter 5. We use the similarity algorithms to measure the distance similarity between numerical values. When a quality request is significantly different from the available Web services in the system or, is far behind the expectation of the available Web services in the system, we use the recommendation based on parameter control algorithms. These are provided by an interface that enables short dialog with the requester to help improve the effectiveness.

Recall that in Chapter 2, we have explained how Hubbard in his book (Hubbard, 2007) emphasized determining the three components of any measurement problem. In our case, we measure the similarity between Web service request and Web service offers quality characteristics. For us, the specified three components are as follows: The concept to be measured is a Web service quality similarity, the object is a part of Web service quality specifications, and the methodology is using distance calculation. Figure 1 demonstrates measurement components of Web service quality similarity. The upper part of the figure
visualizes Hubbard measurement components (Hubbard, 2007); the lower part of the figure demonstrates the Web service quality similarity measurement components.

The following sections presents our work which was published in (Alnahdi et al., 2015). The sections of this chapter are organized as following: Section 4.1 presents the basic idea of the semantic Web service retrieval using simulated data. Section 4.2 presents an overall methodology and experiment. Section 4.3 presents the QoS Ontology. Section 4.4, gives our

![Diagram](image)

**Figure 1. Mapping Hubbard measurement components to Web service quality similarity**

---

1 More background information can be found on my master’s project (Alnahdi, 2011).
proposal of OWL-S Test Set Collection (OWL-S TC) (M. Klusch, Kapahnke, Fries, Alam Khalid, & Vasileski, 2010) extension. The QoS retrieval model is presented in Section 4.5. QoS Parameter Control algorithms are presented in section 4.6. In section 4.7, the tools used in implementing our model are explained. In section 4.8, the experiment of using our model is demonstrated. The results are presented in Section 4.9. Section 4.10 summarizes this chapter.

4.1 Semantic Web Service Retrieval Model

Basically this section introduces a QoS ontology inspired from specifications defined by the World Wide Web Consortium (Lee et al., 2003). The QoS ontology defines relationships among QoS attributes. The QoS attributes conceptualized here are mainly used to define network related characteristics. However, the work can be extended by defining QoS ontologies for other specific domains. Additionally, we extend the test collection OWLS-TC (M. Klusch, Kapahnke, Fries, Alam Khalid, & Vasileski, 2010) according to the QoS ontology. The extension is intended to integrate quantified values for QoS attributes into Web service descriptions. A retrieval algorithm based on similarity measurements is presented. Moreover, deterministic and adaptive parameter control techniques are introduced to guide the requester, when needed, to adjust QoS specifications so better similarity can be achieved. Parameter control algorithms integrate (Pellet, 2010) an OWL reasoner, to reason dependencies among QoS concepts in the QoS ontology.

Web services are software components that enable interoperable interactions over the Internet (Booth et al., 2004). A Web service has an interface that describes its functionality using a descriptive language such as the Web Service Description Language (WSDL) (Christensen, Curbera, Meredith, Weerawarana, & others, 2001). Using this interface description, a Web service can be discovered. However, the proliferation of Web services that provide similar
functionality creates the need to be able to optimally select Web services based on Quality of Service (QoS) requirements, such as performance and availability.

This work proposes a framework for the Semantic Web service discovery through a retrieval algorithm based on QoS requirements. An extensible QoS ontology is introduced to define both domain-independent and domain specific QoS attributes. The ontology is used to comprehensively define the relationships between QoS concepts. The reasoning power of the Semantic Web, explicitly OWL, is used to extract the relationships between two QoS attributes. In addition, the ontology is used to extend the well-known test collection OWL-TC4 (M. Klusch et al., 2010), which includes only functional requirements, so it can include QoS descriptions that can be used for QoS retrieval. The semantic retrieval between a Web service query and offered services is based on the quantified values of the QoS attributes. The core of the framework is the semantic retrieval algorithm that computes the similarity between the values of the QoS attributes in the request and in the functionally equivalent offers. If the similarity retrieval results are low, deterministic and adaptive parameter control algorithms are used to guide the requester to modify her/his query, to achieve superior QoS similarity. In the parameter control process, the Pellet reasoner (Pellet, 2010) is used to reason based on the introduced QoS ontology so that the dependencies among different QoS attributes can be accessed and analyzed. The contributions of this module of the dissertation to the Semantic Web research are: a QoS ontology built based on specification according of the World Wide Web Consortium (Lee et al., 2003), a QoS extension model to the OWL-S_TC (M. Klusch et al., 2010), experiments of similarity measurement and parameter control algorithms for service retrieving, and data mining techniques as a means to compute relevance sets.
4.2 Method and Experiment

The goal of this chapter is to introduce a framework of Semantic Web service retrieval. As seen in Figure 2., this work first introduces a QoS ontology that describes the QoS concepts and the dependencies, represented as OWL properties, among them. This ontology is then used to extend the OWL-S TC (M. Klusch et al., 2010) test collection in a way so that OWL-S descriptions can integrate quantified values for QoS parameters. In addition, a retrieval algorithm, based on measuring the similarity between a service query and offered services, is introduced. Parameter control algorithms are also introduced to recommend adjusting service queries so offered services with even better similarity can be attained. To test the performance of the introduced algorithms, a test set, based on randomly generated QoS values, was created. The test set was loaded into the Weka tool (Hall et al., 2009) to compute the relevance set. Finally, the performance of the algorithms was measured using precision and recall metrics.
4.3 The QoS Ontology

According to ISO (“OECD Glossary of Statistical Terms - Quality - ISO Definition,” 2006), quality is “the totality of features and characteristics of a product or service that bears on its ability to satisfy stated or implied needs.” This chapter introduces a QoS ontology that describes QoS requirements for Web services. The hierarchy of this QoS ontology is based on
Web service requirements introduced by the W3C Working Group (Lee et al., 2003). To create the QoS ontology, Protégé program (*Protege*, 2010) was used to specify the ontology classes, instances (a.k.a. individuals), and properties. Figure 3 shows an overview of the introduced QoS ontology. The main class of this ontology is QoSAttribute, whose instances are QoS parameters such as cost, availability, and execution time. The property hasDescription is used to illustrate the meaning of each QoSAttribute instance. QoSAttribute is divided into DomainIndependentQoSAttribute and DomainSpecificQoSAttribute. DomainIndependentQoSAttribute represents domain independent QoS parameters, including performance and network related attributes. The domain-independent QoS attributes adopted in our ontology of this chapter are based on (Lee et al., 2003). On the other hand, DomainSpecificQoSAttributes is a class that can be extended to define QoS parameters related to a specific domain. The QoS ontology presented here defines QoS attributes related to the traffic monitoring and geographical domains. However, the ontology can be modified and extended to describe QoS parameters in other domains.
The class Type in the QoS ontology is used to classify QoS attributes as categorical or ordinal. A categorical attribute can be represented as a string while an ordinal attribute can be quantified. Instances of the class Metric are the metric units used to measure QoS parameters. The property hasMetric is used to relate instances of the class QoSAttribute to instances of Metric. For example, availability is measured by percentage.

The class Cardinality has two instances, Mandatory and Optional, which are used to represent the importance of a QoS parameter. The property hasCardinality is used to connect instances of the class QoSAttribute with the instances of the class Cardinality. As QoS parameters usually have relationships among themselves, the QoS ontology enables such relationships to be defined. There are two main properties introduced in the ontology: hasDirectRelationship and hasInverseRelationship. hasDirectRelationship is used when the change of one parameter is associated with the proportional change of another parameter in the same direction. For example, the availability has a direct relationship with the cost. On the other hand, hasInverseRelationship is used to represent the relationship between two QoS parameters when one parameter increases and the other proportionally decreases.

Figure 3. QoS Ontology hierarchy using OWL Vis Tab in Portege
4.4 OWL-S TC Extension

This chapter uses OWL-S TC4 (M. Klusch et al., 2010) as a test set. Since OWLS TC4 is designed to evaluate the performance of functional retrieval algorithms but not QoS retrieval algorithms, this chapter extends OWL-S 1.1 descriptions of the test set for the geographical domain. The extension process is shown in Figure 4. At first, the OWL-S TC (M. Klusch et al., 2010) is installed in an Apache Friends (Apache Friends, 2010) and OWL-MX MX (Matthias Klusch, Fries, & Sycara, 2009) is used to do functional retrieval and query about Web service offers that match a query in the geographical domain. The OWL-S description for the query and the resulting matched services are then extended to include the service category “geographical” using the serviceCategory property of the OWL-S profile as shown in Figure 4. Services that have the same categories are considered functionally equivalent.

Figure 4. Extension process of OWL-S TC4
In addition, extensions for QoS parameters are added to OWL-S descriptions. The OWL-S service parameter tag is extended to include quantitative information of QoS parameters, as shown in Figure 5. This information includes a range represented by a minimum value, a maximum value, and the average value. The values of these attributes were generated by a random value generator. Each value was computed as the average of 20 random values.

![Extension of ServiceCategory and ServiceParameter that includes quantitative values for QoS parameters](image)

**Figure 5. Extension of ServiceCategory and ServiceParameter that includes quantitative values for QoS parameters**

### 4.5 QoS Retrieval Model

The QoS extended model represents QoS parameters of both queries and offers using numerical values. Because each service (query or offer) is represented by several quantified values for corresponding QoS parameters, a service can be considered a Euclidean geometrical point represented by several dimensions. A simplified example is demonstrated in Table 1. According to the notion of Web services as geometrical points, the distance between a query and an offered service may be represented by the similarity between them. A lower value of the
similarity represents a higher or better matching. The core of the retrieval process in this chapter is based on the Minkowski distance shown in the following formula as shown in Equation 1.

**Equation 1. Minkowski Distance**

\[
Minkowski \ Distance = (\sum_{i=1}^{n}|x_i - y_i|^p)^{1/p}
\]

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Cost</th>
<th>Response Time</th>
<th>Reliability</th>
<th>Accuracy</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Query</td>
<td>40</td>
<td>4</td>
<td>98</td>
<td>78</td>
<td>34</td>
</tr>
<tr>
<td>Offer 1</td>
<td>10</td>
<td>1</td>
<td>78</td>
<td>78</td>
<td>23</td>
</tr>
<tr>
<td>Offer 2</td>
<td>43</td>
<td>7</td>
<td>56</td>
<td>56</td>
<td>64</td>
</tr>
<tr>
<td>Offer 3</td>
<td>98</td>
<td>3</td>
<td>43</td>
<td>34</td>
<td>34</td>
</tr>
<tr>
<td>Offer n</td>
<td>12</td>
<td>2</td>
<td>88</td>
<td>69</td>
<td>34</td>
</tr>
</tbody>
</table>

The choice of the Minkowski distance, a general form of Euclidean distance, was based on the consideration of Web services as Euclidean geometrical points. The algorithm of the retrieval process is demonstrated in Figure 6. First, the functional retrieval is based on the serviceCategory value of both the query and the offered services. Next, QoS values of the query are extracted into an array, and QoS values of the offers are extracted into a matrix. After that, the similarity between the query and each offered service is calculated based on the Minkowski
distance to generate a similarity array. Finally, the results are ranked and a threshold (in this case 40%) of the results is retrieved. The choice of retrieving 40% of the services was based on the average value of the number of services of the relevance set when applying a clustering algorithm on several datasets.

![Figure 6. QoS retrieval based on minkowski distance](image)

### 4.6 Parameter Control

When the retrieval algorithm is applied to measure the similarity between a query and a set of offered services, the similarity results might not be satisfying (*e.g.*, none of the results exceeds a given threshold). Therefore, a suggestion to modify the QoS values of the query can be given to the requestor so that the retrieval process might result in a better performance (*e.g.*, a higher similarity result). The suggestion can be done through the parameter control. In this chapter, two types of parameter controls from the domain of parameter controls in evolutionary algorithms (Eiben, Hinterding, & Michalewicz, 1999) are explored. First, a deterministic parameter control in which each step (*i.e.*, the factor that is used to modify each QoS value) is

---

**Algorithm 1** Web Service Retrieval

```plaintext
1: procedure WebServiceRetrieve(Query, Offers)
2:     Qsc ← serviceCategory of Query
3:     FRO ← OffersWithServiceCategoryQsc  ▷ FRO Functionally Retrieved Offers that is equivalent to the query
4:     QoSQueryArray ← extractedQoSParametersValuesOfQuery
5:     QoSofferMatrix ← extractedQoSParametersValuesOfFRO
6:     o ← 0
7:     while o < QoSofferMatrixSize do
8:         Sim[o] ← MinkowskiDistance(Query, QoSofferMatrix[o])
9:         o ← o + 1
10:     end while
11:     Results ← RankQMatrixBasedOnSim
12:     return Top T of Results  ▷ T is a threshold
13: end procedure
```

---
defined by a predefined math formula and changes in time. Second, an adaptive parameter control, in which the step is not only defined based on time but also based on the result of the similarity, gives feedback to the process. For example, if the similarity is bad at the beginning of the iterations, larger steps can be considered. If the similarity is good at later phases of the iterations, smaller steps can be considered.

When changing the values of QoS parameters during the parameter control cycles, the relationships between QoS parameters have to be considered. For example, when increasing the value of the cost parameter, the value of availability has to be increased accordingly. This is because the cost and availability are related by the hasDirectRelationship property in the QoS ontology. On the other hand, QoS parameters that are related by the hasInverseRelationship property in the QoS ontology, such as cost and execution time, are changed inversely. Namely, when the value of cost increases, execution time decreases proportionally. For this reason, at the beginning of the parameter control process, the relationships are computed using the Pellet reasoner to identify the parameters that have hasDirectRelationship and hasInverseRelationship properties. Figure 7 shows the code used to perform the property reasoning. The method Has Positive Relation has two parameters that represent the QoS individuals (a.k.a instances) that need to be checked to determine if they have the direct relation dependency. At first, the method creates an ontology model to read the QoS ontology with the support of the Pellet reasoner. Then, it retrieves the individuals that have hasDirectRelationship with the first parameter and returns true if the second parameter does, in fact, have a direct relationship with the first parameter. Otherwise, the method returns false. Parameter control to utilize either a deterministic approach as shown in Figure 8 and an adaptive approach Figure 9 to adjust the “stride” of a suggested query. As seen in Figure 8 at the beginning, the strategy array is initialized by the
Then, the Is Direct (ISD) Boolean array is computed using the hasDirectRelation method to indicate the dependency between the first QoS parameter and other parameters. Then the average of the first QoS parameters in the offer matrix is calculated to guide the direction of the strategy (increase or decrease). Then, according to the ISD value and the strategy value in each iteration, the suggested query is computed Figure 9 demonstrates the adaptive parameter control algorithm that is similar to the deterministic parameter control algorithm. The main difference is that the adaptive algorithm depends in each iteration on the feedback of the similarity between the suggested request and the available offer. If the similarity exceeds the threshold, then the adjusting of the QoS parameters will be based on a bigger step; otherwise it will be based on a smaller step.

![Algorithm 2 Has Positive Relation](image)

**Figure 7. Property reasoning**
Figure 8. Deterministic approach Web service retrieval
Algorithm 4 Web Service Retrieval Based on Adaptive Tuning

1: procedure AdaptiveTuningWsRetrieval(query, offers)
2:   mti := 10  \( \triangleright \) mti= max tuning iterations
3:   fpavg := 0.0 \( \triangleright \) first parameter average
4:   smallStep := 0.1
5:   bigStep := 0.3
6:   badSim = 1000
7:   suggestedQuery := query
8:   strategy := initializeStrategyArrayWith(1) \( \triangleright \) strategy array
9:   ISP := ComputeIsPositiveRelation() \( \triangleright \) ISP is array of boolean,
  Compute is positive relation using property reasoning
10:  sim := webServiceRetrieve(query, offers)
11:  minSim := sim[0]
12:  i := 0
13:  while (OffersHaveElements) do
14:    fpavg := fpavg+offers[i][0] \( \triangleright \) compute the first parameter average
15:    i := i+1
16:  end while
17:  fpavg := fpavg/numberOfOffers
18:  step := 0
19:  j := 0
20:  while (j < mti) do
21:    if (minSim < badSim) then
22:      step := bigStep
23:    else
24:      step := smallStep
25:    end if
26:    if (request[0] > fpavg) then
27:      strategy[0] := (strategy[0]/1)+step
28:    else
29:      strategy[0] := (strategy[0]+step)
30:    end if
31:  end while
32:  suggestedQuery[0] := suggestedQuery[0]*strategy[0]
33:  p := 1
34:  while (p < queryLen) do
35:    if ISP(p) then \( \triangleright \) If it has positive relation
36:      strategy[p] := strategy[0]
37:    else
38:      strategy[p] := 1/strategy[0]
39:    end if
41:    p := p+1
42:  end while
43:  sim := WebServiceRetrieve(suggestedQuery, Offers)
44:  minSim := average(sim)
45:  return suggestedQuery
46: end procedure

Figure 9. Adaptive approach Web service retrieval
4.7 Tools

- Weka

Weka (Hall et al., 2009) is a free open source tool that provides a collection of machine learning and data mining algorithms. The algorithms can be applied directly to a dataset using Weka’s graphical or command line interfaces, or they can be accessed from Java code. In addition, Weka contains tools for pre-processing, data cleaning, classification, clustering, regressions, association rules, and visualization. Weka can also be used for developing new machine learning algorithms. On the one hand, Weka’s graphical interface is intended to be used for initial experiments. On the other hand, Weka’s command line interface is good for in-depth usage because it provides functionalities that are not available in the graphical interface and it uses less memory. A dataset in Weka is equivalent to a two-dimensional spreadsheet or a database table. Weka implements a dataset by the class Instance. A dataset consists of a group of examples, each of which is an object of class Instance. Each instance can be of type nominal (predefined value), numeric (integer or real number), or string (list of characters delimited in double quotes). The representation of Instance is an Attribute Relationship File Format (ARFF) file. Each ARFF file contains a header that describes the relation name and attributes types and has the token @data followed by the dataset as a comma-separated list. By default, the last attribute is considered as the class attribute (which is the attribute that can be predicted as a function of all other attributes). Any machine learning algorithm in Weka is derived from the class Classifier. A classifier model is based on applying a complex mapping to all but one attribute. The performance of a classifier can be evaluated by the accuracy (error rate) based on the proportion of correctly predicted examples to all examples in an unseen dataset. Another way is to use the hold-out estimate method, which is based on using a training set and a test set that
are independent. To estimate the variance in these performance estimates, holdout estimates can be computed by repeatedly reassembling the same dataset and computing the average and standard deviation of accuracy.

- **OWL-MX**

OWL-MX (Matthias Klusch et al., 2009) is a hybrid OWL-S matchmaker. It is used to retrieve services for a given query with both written in OWL-S and based on the OWL language recommended by W3C. OWL-MX provides pure profile Input-Output matchmaking combined with logic-based semantics and with syntactic token-based similarity metrics in order to take advantage of both description logic and information retrieval domains. OWL-MX was developed at the German Research Center for Artificial Intelligence (“German Research Center for Artificial Intelligence,” 2017) as the first hybrid matchmaker available in the Semantic Web community. OWL-MX is written in Java and uses the of Pellet reasoner for logic based filtering.

### 4.8 Experiment

- **Creating Test Set**

This experiment uses a random generator module to create random values for Web service offers that involve 5 different QoS parameters (cost, execution time, availability, reliability, and accuracy). The choice of these parameters allows the experiment to include different dependencies between QoS parameters. For example, cost has a direct relationship with availability and an inverse relationship with execution time. The random generator module automatically creates an ARFF file.
• Creating Relevance Set

A relevance dataset is the dataset that is relevant to the query. Creating a relevance set in the Web service functional retrieval domain requires a human expert to decide the binary or graded relevance between a service request and a service offer (Küster & König-Ries, 2010). However, in non-functional retrieval there is no standard method for creating relevance sets. Therefore, this chapter proposes using unsupervised data mining algorithms for relevance set creation. To create a relevance set, the ARFF dataset that is generated using the generator module is loaded into the Weka tool (Hall et al., 2009). The dataset consists of the relation name (table name) followed by the attribute names, each of which is followed by the designated type. Then, the values are represented in a Comma Separated Value (CSV) format. The first instance represents the query and the other instances represent offers. The SIB clustering algorithm is used to cluster the query and service offers. The choice of SIB was based on its good performance in document clustering applications (Ji & Ye, 2011).

• Test Collection

OWL-S TC4 (M. Klusch et al., 2010) is an OWL-S service retrieval test collection. The purpose of this test collection is to measure the performance of OWL-S matchmaking algorithms. The test collection provides 1083 Web services written in OWL-S 1.1. These Web services are from different domains (e.g., communication, economy, education, geography, food, medical care, simulation, and travel). Some of the services are from public registries such as IBM UDDI registries. Semi-automation methods were used to translate WSDL Web services to OWL-S. In addition to the Web services, the test collection provides 42 test queries, which are associated with relevance sets, both binary and graded, for evaluating experiment performance. The relevance sets were created using the Semantic Web Service Relevance Assessment Tool.
(SWSRAT) (Cabral & Toma, 2010). The relevance sets are based on a scale with 4 values: highly relevant (3); relevant (2); potentially relevant (1); and non-relevant (0).

- **Performance Evaluation**

Since Web service selection is an information retrieval problem, this chapter evaluates the performance of the matchmaking algorithm using the widely accepted metrics, precision and recall, from the information retrieval domain. Precision and recall are defined in terms of retrieved services (services that are returned by the retrieval algorithm) and relevant services (services that are relevant to the query). Precision is the ratio of relevant retrieved services to all retrieved services, and it measures how well the retrieval algorithm weeds out the unwanted results. Recall is the ratio of the retrieved relevant services to all relevant services, and it measures how well the retrieval algorithm finds the desired results. 10 experiments were conducted to evaluate the retrieval algorithm, and the number of QoS offers was increased in each experiment.

### 4.9 Results

As can be seen from Figure 10 (a), when the size of offers is between 40 and 70 Web services, the three algorithms (*i.e.*, parameter tuning, deterministic, and adaptive) behaved competitively. However, the deterministic parameter control algorithm performs best when there are more than 80 offered services. Regarding the precision shown in Figure 10(b), the performance of the matchmaking algorithm (without parameter control) is better than the performance of the adaptive parameter control when the size of offers is between 40 and 70. In general, the deterministic parameter control algorithm has better precision than the retrieval algorithm without parameter control and the adaptive parameter control algorithm. The
experiments show that when there are more offers, adjusting QoS requests using the deterministic parameter control algorithm results in the discovery of more suitable offers with smaller similarity.

![Figure 10. Precision and recall](image)

4.10 Summary

This chapter contributes to the Web service retrieval by providing a QoS ontology which is based on the W3C (Lee et al. 2003) specifications of the QoS parameters and which is an extended model of the OWL-S TC test collection. An experiment that evaluates the performance of the retrieval algorithm based on similarity measurements between a requested service and a set of service offers is the parameter control algorithm, which advises the requester to adjust the query so better similarity can be achieved. For future work, we plan to investigate methods other than data mining techniques to find a suitable way of measuring the relevance set that is used in evaluating the Web service retrieval performance. Further improvements of parameter control algorithms will be investigated to enhance performance.
CHAPTER 5

Web Service Retrieval Based on Quality a Case Study

Quality Based Web Service Information Retrieval Systems (QBWSIRS) are used to help Web service users to retrieve and select Web services based on users’ specifications. Furthermore, when retrieving functionally equivalent Web services, Web service users can filter their selections based on quality. Chapter 4 provided a proposed model for QBWSIRS on a simulated dataset created by randomizing data values. In this chapter, we test our model in Chapter 4 on a different dataset. In this chapter, we use a real quality dataset and see how it affects the training of our model. The similarity between a Web service query and a Web service offer is calculated by applying distance equations such as the Minkowski distance. The retrieval model is used to help users to select Web services based on specified qualities.

5.1 Dataset Description

The test data used in this experiment are a standard dataset provided by (Al-Masri & Mahmoud, 2007a) (Al-Masri & Mahmoud, 2007b) (Al-Masri & Mahmoud, 2008a). The dataset is stored in a plain text file for simplicity purposes. The rows of the dataset represent quality characteristic values of a Web service. The dataset consists of numerical and string values. This dataset consists of 13 attributes, eleven of which are numerical quality attributes that describe quality of a service such as throughput and availability, service name, and WSDL address. The Web services were carefully selected by the authors (Al-Masri & Mahmoud, 2007a) (Al-Masri & Mahmoud, 2007b) (Al-Masri & Mahmoud, 2008a) so they would represent several domains and functionalities. Each row in the dataset consists of 13 values describing a real Web service, and each value represents the average of collected values over the collecting time period. Table 2
describes data information by identifying quality parameters, descriptions of those parameters, and measurement units. The definitions of these quality parameters are specified in Table 2 as defined in the dataset Website Quality Dataset publication and Website (Al-Masri & Mahmoud, 2008). Types of these parameters include ordinal, categorical, and string categories. Parameters from 1 to 11 are quality domain independent characteristics that can be shared between all kinds of services. WsRF and service classification are computed values by the authors of the QWS dataset to classify and rate Web services based on their quality. Service Name is a parameter of string type can be used as an indicator of the function of the associated Web service. WSDL Address is used as the identifying parameter in our model because it refers the user to the unique network address of the Web Service.
## Table 2. QWS dataset description

<table>
<thead>
<tr>
<th>Parameter Type</th>
<th>Quality Parameter Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>Attribute</td>
</tr>
<tr>
<td>1</td>
<td>Response Time</td>
</tr>
<tr>
<td>2</td>
<td>Availability</td>
</tr>
<tr>
<td>3</td>
<td>Throughput</td>
</tr>
<tr>
<td>4</td>
<td>Successability</td>
</tr>
<tr>
<td>5</td>
<td>Reliability</td>
</tr>
<tr>
<td>6</td>
<td>Compliance</td>
</tr>
<tr>
<td>7</td>
<td>Best Practices</td>
</tr>
<tr>
<td>8</td>
<td>Latency</td>
</tr>
<tr>
<td>9</td>
<td>Documentation</td>
</tr>
<tr>
<td>10</td>
<td>WsRF</td>
</tr>
<tr>
<td>11</td>
<td>Service Classification</td>
</tr>
<tr>
<td>12</td>
<td>Service Name</td>
</tr>
<tr>
<td>13</td>
<td>WSDL Address</td>
</tr>
</tbody>
</table>
5.2 Dataset Normalization

The quality dataset used in this dissertation contains parameters of different units and scales. When calculating those parameters together in the same equation it is essential to normalize the data. Distance equations such as Manhattan distance, Euclidean distance, or Minkoski distance equation has n space. The issue is how to use different types of measurements in the same equation. Dealing with a parameter such as response time which has millisecond (ms) as its unit is different than when dealing with reliability which has percentage (%) as a unit, because the jump from 12 ms to 20 ms is not the same as the jump of 12% to 20%. When performing statistics and distance measurements, the issue of doing calculation of arguments that have different units and different scales has to be addressed. One way to handle calculation of parameters that have different units and scales is to do normalization for the data. Data normalization as described in (DRKPI, 2009) is a way of handling composite variables made up with different units and putting those values into a unified unit. One way of transforming parameters into the same scale is standardization (DRKPI, 2009) that makes the average zero and standard deviation one. Equation 2 shows the normalization methodology used in the experiment.

\[
\text{Score Normalization} = \frac{\text{value} - \text{mean of sample values}}{\text{standard deviation of sample values}}
\]
For our calculations, we begin with a z-score formula, but in place of the population mean and the population standard deviation, we are use a sample space mean and a sample space standard deviation, respectively. Thus, our results only approximate completely rigorous statistics. However, we do believe that our results do indicate the potential merit of our efforts.

Figure 11 shows an example of normalizing Web service quality values of QWS dataset V1. The top part of the figure shows a table that illustrates the original dataset before normalization. The head of the table shows quality attributes (columns) of the dataset Quality Attributes: Response Time (RT), Availability (AV), Throughput (TH), Successability (SC), Reliability (RL), Compliance (CM), Best Practice (BP), Latency (LT), Documentation (DC), Web Service Relevancy Function (WSRF), Service Classification (SCL). The first two rows show Average (AVG) and Standard deviation (SD) values of the eleven quality attributes values. Other rows show quality values of the dataset of several Web services. The bottom part of the figure shows the data after normalization.
**Figure 11. Applying normalization to a dataset**

QWS Dataset Before normalization

<table>
<thead>
<tr>
<th>QA</th>
<th>RT</th>
<th>AV</th>
<th>TH</th>
<th>SC</th>
<th>RL</th>
<th>CM</th>
<th>BP</th>
<th>LT</th>
<th>DC</th>
<th>WSRF</th>
<th>SCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVG</td>
<td>840.28</td>
<td>84.75</td>
<td>7.28</td>
<td>64.43</td>
<td>61.31</td>
<td>83.67</td>
<td>80.68</td>
<td>763.48</td>
<td>47.53</td>
<td>66.65</td>
<td>2.78</td>
</tr>
<tr>
<td>SD</td>
<td>2764.31</td>
<td>20.45</td>
<td>6.45</td>
<td>21.17</td>
<td>21.19</td>
<td>8.77</td>
<td>6.69</td>
<td>2755.55</td>
<td>36.40</td>
<td>11.50</td>
<td>0.98</td>
</tr>
<tr>
<td>WS1</td>
<td>45</td>
<td>83</td>
<td>27.2</td>
<td>50</td>
<td>97.4</td>
<td>89</td>
<td>91</td>
<td>43</td>
<td>58</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>WS2</td>
<td>71.75</td>
<td>100</td>
<td>14.6</td>
<td>88</td>
<td>85.5</td>
<td>78</td>
<td>80</td>
<td>64.42</td>
<td>86</td>
<td>93</td>
<td>1</td>
</tr>
<tr>
<td>WS3</td>
<td>117</td>
<td>100</td>
<td>23.4</td>
<td>83</td>
<td>88</td>
<td>100</td>
<td>87</td>
<td>111</td>
<td>59</td>
<td>90</td>
<td>1</td>
</tr>
<tr>
<td>WS364</td>
<td>1680.16</td>
<td>16</td>
<td>2</td>
<td>7</td>
<td>12.5</td>
<td>89</td>
<td>72</td>
<td>1667.22</td>
<td>11</td>
<td>30</td>
<td>4</td>
</tr>
</tbody>
</table>

QWS dataset after normalization

<table>
<thead>
<tr>
<th>QA</th>
<th>T</th>
<th>AVR</th>
<th>TH</th>
<th>SC</th>
<th>RL</th>
<th>CM</th>
<th>BP</th>
<th>LT</th>
<th>DC</th>
<th>WSRF</th>
<th>SCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS1</td>
<td>-0.29</td>
<td>-0.09</td>
<td>3.08</td>
<td>-0.68</td>
<td>1.70</td>
<td>0.61</td>
<td>1.54</td>
<td>-0.26</td>
<td>0.29</td>
<td>2.90</td>
<td>-1.82</td>
</tr>
<tr>
<td>WS2</td>
<td>-59.83</td>
<td>0.96</td>
<td>0.30</td>
<td>1.34</td>
<td>0.66</td>
<td>0.78</td>
<td>0.81</td>
<td>-62.58</td>
<td>0.86</td>
<td>0.81</td>
<td>0.02</td>
</tr>
<tr>
<td>WS3</td>
<td>1.00</td>
<td>0.17</td>
<td>-0.26</td>
<td>0.38</td>
<td>-0.11</td>
<td>0.14</td>
<td>-0.05</td>
<td>1.06</td>
<td>0.01</td>
<td>-0.11</td>
<td>0.00</td>
</tr>
<tr>
<td>WS364</td>
<td>0.20</td>
<td>0.07</td>
<td>2.00</td>
<td>-0.14</td>
<td>-0.02</td>
<td>-0.14</td>
<td>-0.04</td>
<td>0.21</td>
<td>2.00</td>
<td>-0.13</td>
<td>0.20</td>
</tr>
</tbody>
</table>
5.3 Dataset Distribution

To illustrate the nature of the dataset used in our model, we are looking to see how data are distributed. To demonstrate the distribution of values of the Web service quality dataset (Al-Masri & Mahmoud, 2007a) (Al-Masri & Mahmoud, 2007b) (Al-Masri & Mahmoud, 2008a) data, we used a Weka software (Weka, 2016). The results are given in Figure 12. The attributes bestPractices, WsRF, and reliability follow a normal distribution.

<table>
<thead>
<tr>
<th>Response Time</th>
<th>Availability</th>
<th>Throughput</th>
<th>Sccessability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reliability</th>
<th>Compliance</th>
<th>BestPractices</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Documentation</th>
<th>WsRF</th>
<th>ServiceClassification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 12. Web Service quality dataset distributions**

5.4 Experiment

Our experiment of measuring similarity between a Web service query and Web service offers was performed on the data in two formats non-normalized and normalized. The details of the two experiments are as follows:
The implementation of the proposed retrieval approach is done by a C++ program which considered a query to be the first line in the dataset. Retrieval quality modules compare a quality query with available dataset offers as provided by the dataset explained previously in this chapter. The module is based on a Web service quality class in the program that defines the parameters provided in Table 2. The main class of the program is swsq (Semantic Web Service Quality). Data members of this class are based on attributes defined by the quality dataset in Table 2. The swsq has 13 data members specifically (response time, availability, throughput, successability, reliability, compliance, Best Practice, latency, Documentation, WsRF, Service Classification, Service Name, WSDL Address).

Member functions of the class swsq include setters to set data values to the data members and getters to retrieve data values for data members. In addition, class swsq consists of an uploading function to upload the quality dataset values from a file to the memory of the program. The MinkowskiDistanceCalculator function calculates the similarity between two Web services quality descriptions. In addition, the scaledMinkowskiDistanceCalculator function computes the distance between a Web service query and several Web service offers. Moreover, the readQuery() function is a function that is responsible for reading query values from a user. printRankedResults is a function that prints Web services based on their relevance to the Web service query. Display Web service quality is a function that is responsible to print a Web service quality values on the screen. Test classes are used for testing the accuracy of the functionality for the functions in the program.

Techniques for calculating similarity include distance calculations methodologies. The choice of distance calculation equations is based on envisioning quality characteristics of a query as a geometric point on the space and quality characteristics of Web service offers as other points
that also can be seen as geometric points. In the experiment, a query point is a point with several dimensions, each of which represents a quality value. The similarity distance is calculated between the query and an offer in the dataset. An offer that has a good similarity value will rank higher in the result.

5.5 Selecting Algorithms

The steps of our algorithm are demonstrated in Figure 13. To measure the similarity between a query and an offer is to estimate the distance between them by quantifying that distance to a numeric value. The independent variables of the dataset that are used are the first 11 quality parameters as shown in Table 2. The dependent parameter is the similarity result calculated according to Minkowski distance equation. The smaller the value of the similarity between a Web service query and a Web service offer, the more similar the offer is to the query.

![Algorithm 1 Web Service Retrieval Algorithm](image)

**Figure 13. Web service retrieval based on quality**

The implementation of the proposed methodology is done by a C++ program, as explained earlier in this chapter, which defines a Web Service Quality class that has the quality parameters specified in Table 2. The program uploads the specified quality dataset into a set of vectors; one vector for the request and another vector for each offer. Retrieval quality modules
compare a quality query (in this case, the first quality values from the dataset) with available offers by calculating the similarity. Then, the program ranks the results based on their similarity to the query. In the experiment, we assumed that the first Web service quality data in the dataset to be the query and other data are available offers. Figure 14 demonstrates the ranked results of Web services based on quality.

5.6 Experiment Evaluation

We used the well-known information retrieval measures precision and recall, Equation 3 and Equation 4 respectively, to evaluate our results. The precision and recall are used to compare the results of our program and the results given by clustering techniques in Weka (Weka, 2016). We assumed that the results of Weka would be our relevant values.

Figure 14. Web service ranked results
The clustering technique that we have used in Weka program (Weka, 2016) is Make Density Based Cluster. We considered the relevant set to be the set that is produced by using the clustering algorithm specifically MakeDensityBased that is implemented by Weka program (Weka, 2016). The selection of Make Density clustering due to its several benefits including its ability to handle noise data (Moreira, Santos, & Carneiro, 2005). Figure 15 shows the result for clustering the QWS1 dataset using Weka (Weka, 2016). Figure 17 shows clustering model in Weka. When clustering using Weka, serviceName and WSDLAddress attributes were ignored since they are only functional and identification characteristics and have no indication of the quality similarity. In addition, Figure 16 shows the clustering assignment results.
Figure 15. QWS1 Clustering visualization result using weka
Figure 17. Clustering model using MakeBasedDensity clusterer

Figure 16. Clustering assignment for QWS file in Weka QWS
5.7 Results and Discussion

We computed precision and recall for both the experiments with qws1 with no normalization and qws1 with normalization; the results are in Table 3 and Table 4. They show that precision value is better than recall, and that normalizing a dataset will enhance the precision and recall results which increase the accuracy of Web service similarity calculation.

Table 3. Precision and recall for original QWS1 dataset

<table>
<thead>
<tr>
<th>Retrieved Web services</th>
<th>146</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevant Web services</td>
<td>196</td>
</tr>
<tr>
<td>Relevant and Retrieved Web services</td>
<td>107</td>
</tr>
<tr>
<td>Precision</td>
<td>0.7329</td>
</tr>
<tr>
<td>Recall</td>
<td>0.5459</td>
</tr>
</tbody>
</table>

Table 4. Precision and recall for normalized QWS1 dataset

<table>
<thead>
<tr>
<th>Retrieved Web services</th>
<th>146</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevant Web services</td>
<td>196</td>
</tr>
<tr>
<td>Relevant and Retrieved Web services</td>
<td>126</td>
</tr>
<tr>
<td>Precision</td>
<td>0.8631</td>
</tr>
<tr>
<td>Recall</td>
<td>0.6429</td>
</tr>
</tbody>
</table>
5.8 Summary

Web services are software modules that provide functional service over the Internet. These Web services might have different quality characteristics that describe them. Differentiating between functionally equivalent Web services according to Quality is essential when there are multiple Web services that perform the same functionality. This chapter provided a measuring model for estimating the information retrieval quality of Web services. To train our model, we used a real quality dataset.

This chapter provided a proposed quantitative approach for studying quality retrieval and ranking for Web service information retrieval systems. The dataset used to train our algorithms is described. Also, a detailed implementation of retrieval and ranking module are described. We adopted a quantitative methodology for studying these quality factors. Specifically, studying quality factors will impact the design of building a Web service retrieval system based on quality. The scalability of the selection algorithm can be managed by using the well-known big data model specifically map and reduce paradigm (Dean & Ghemawat, 2008). Hadoop (Lam, 2010) can be used to implement the map and reduce paradigm.
CHAPTER 6

Quality Function Deployment (QFD) and Maintaining the Quality of Web Service Description Artifact

In this chapter, we investigate how House of Quality (HoQ) can be used when it comes to the Web service domain was investigated. This chapter also represents our work in (Alnahdi, Melton, & Liu, 2016). HoQ, as demonstrated in Chapter 2, used in industrial domain to reduce design time while maintaining or improving quality. In the previous chapter, it was mentioned how people used HoQ and Quality Function Deployment (QFD) in the computer science and software domain. In this chapter, we describe our perspective on the QFD when it comes to Web service description artifact.

The artifact that we are primarily concerned about is the Web service description for Web services. Web service description resembles a resume for a Web service. Having a well written Web service description is essential when publishing that Web service in a Web search repository and making it available for retrieval. Including detailed descriptions of both functional and non-functional (e.g. quality) specifications of a Web service description will increase the chances of the Web service being selected.

As in chapter 2, we mentioned that the original HoQ concept is concerned with listening to the voice of the customer and the voice of the engineer. In like manner, we expect that listening to the stakeholders involved in a Web service domain is essential. In a Web service domain, stakeholders include Web service provider, Web service requester, and Web service
registry holders. Every single party of these stakeholders has some requirements, expectations, and perspective about quality characteristics of Web service and how they are described.

Web service quality characteristics can be classified into two categories: domain independent, and domain specific characteristics. Domain Independent characteristics are network quality descriptions and other features which shared by all Web services, since all Web services are provided through a network connection. These characteristics can be learned from listening to network experts. Domain specific characteristics are used to describe quality information about a specific domain, such as educational, marketing, or accounting. Identifying domain specific quality characteristics can be achieved by listening to experts in those domains. The voice of engineers in general and for specific domains is essential in identifying quality characteristics from the designers’ point of view.

The voice of Web service users or selectors is also essential. Users have different perspectives of what quality means. Listening to what users need, want, or may want is essential in building a quality model for Web services. Identifying different grades of quality requirements is essential in prioritizing and weighting those requirements when selecting Web services. What is mandatory or essential for one person might only be optional to another user. Identifying such quality differences is essential in building Web service retrieval systems based on quality.

In addition to the voice of Web service engineer and Web service user, the voice of Web service owner also has to be heard when identifying quality characteristics. Owners of Web services characterize quality of Web services based on their ability on what they can provide. For example, pricing, response time, and speed are some of quality characteristics that concern Web service owners. Having these quality specifications in a Web service description will enhance the quality of the Web service description by including valuable content.
Furthermore, listening to Web service repository moderator vision of quality characteristics of a Web service is critical. Web service repositories include descriptions information about manifold Web services. These Web services vary in their domain. Web service users and providers might have a narrative perceptive of quality characteristics based on their interest of a specific kind of Web service. However, we expect moderators of Web service repositories to have a comprehensive conception by virtue of their work of handling variety of Web services that fit to different domains. We anticipate this broad vision to enable Web service moderators to characterize general quality characteristics that are common between Web services. On the other hand, repository moderators would be able to determine what quality characteristics that is specific to special domains. Web service moderators have the accessibility to Web service repositories and comparing those Web services and different Web service providers and understanding the behavior of publishing and selecting. This kind of ability qualifies them to recognize competitive quality characteristics of Web services. Also, we expect the Web service moderators to be able to define and standardize quality characteristics of Web services.

In Chapter 2, the three measurement components are reviewed as specified in (Hubbard, 2007), concept, object, and methodology. This chapter provides measuring the quality of Web service description, the artifact to be measured is Web service description, and the methodology is WSDQFD. Figure 18 demonstrate these three components proposed by (Hubbard, 2007) and the corresponding components in the domain of the quality of Web service description artifact.
As can be seen from Figure 18, the concept to be measured is the quality of Web service description. The quality of Web service description can be maintained by asserting valuable content in the Web service description. This can be done by the proposed Web Service Description Quality Function Deployment (WSDQFD) in this dissertation. The steps of applying WSDQFD are further illustrated in Figure 19.

Figure 18. Measurement components of applying QFD to Web service description

As can be seen from Figure 18, the concept to be measured is the quality of Web service description. The quality of Web service description can be maintained by asserting valuable content in the Web service description. This can be done by the proposed Web Service Description Quality Function Deployment (WSDQFD) in this dissertation. The steps of applying WSDQFD are further illustrated in Figure 19.
We propose systematic procedural steps for assuring quality of Web service description artifact. These steps are demonstrated in Figure 19. The details of each step are demonstrated as following:

### 6.1.1 Collecting Web Service Quality Requirements

The beginning of this Chapter talks about different stakeholders who have different viewpoints of quality of a Web service description artifact, including Web service engineers, Web service providers, Web service requesters, and Web service repository moderators. Listening to those stakeholders would generate many of quality characteristic specifications.
Web service quality description artifact moderator would be responsible to manage the process of collecting quality specifications from Web service quality stakeholders. This will enable the moderator to combine different quality requirements from different perspectives into quality manuscripts so these requirements will be further processed by the next step of WSDQFD. Requirement collection can be captured by in-person collaborating meetings, online meetings, emails, online forums, and online questionnaire forms. Online systems of collecting information would be easier to classify quality attributes and requirements.

Quality domain-specific attributes can be collected from people who are experts and specialized in a specific domain. Those people would be familiar with the jargon used to describe quality of Web services and identify the actual definition of those qualities. Having appropriate jargon to describe quality factors is essential to creating accurate Web service description.

6.1.2 Filtering Web Service Quality Requirements

The previous section describes how the data can be collected from different stakeholders. Once the data have been collected, there will be redundant information due to overlapping expectations from stakeholders. Filtering those overlapped quality expectations by eliminating redundancy while keeping corresponding frequency numbers is essential. The filtering process can range from a manual, semi-automatic, or fully automatic process. When specifications are limited and collected through meetings, manual filtering would be suitable. Automatic filtering is appropriate when quality data is collected through online procedures. The intervention of a human being to monitor and review the filtering process may be helpful even essential to ensure the quality of the filtered data.
6.1.3 Prioritizing Web Service Quality Requirements

After filtering quality requirements by eliminating redundancy, there will be a need to organize and prioritize those needs based on importance of the quality requirements. A highly important quality value may be the one that repeated frequently when collecting values. Also, quality requirements that have been proposed from experts would have high priority. Identifying highly selected quality requirements is essential in selecting the final priority parameters to be included in the Web service description artifact. Prioritizing according to frequency of quality data can be done automatically. However, manual prioritizing for the final drafts is expected to validate the automatic prioritizing process.

6.1.4 Classifying Web Service Quality Requirements

After deciding what quality requirements to be included in the Web service quality description, classifying those Web services according to domain-independent and domain-specific is mandatory to separate the general quality requirements from the specific requirements in different quality groups. Domain specific requirements can be further classified into different domains, and subdomains. Other classification hierarchies are also beneficial when structuring data in the Web service description artifact. More detailed classifications are significant for Web service retrieval systems that allow sophisticated search queries be handy to Web service users. Initial classification of quality requirements can be done automatically by using clustering techniques. Manual classifying on the automatically classified data to find the relationship between those concepts can be done by an analyst who has read specification manuscripts and knows the relationships between quality concepts.
6.2 Model Evaluation

To verify our proposed model (WSDQFD) of using systematic approach involves several steps, as illustrated in Figure 19 for creating a quality Web service description artifact, an online survey was conducted to ask researchers who are experts in the field to evaluate the methodology. Survey participants are researchers in the domain of computer science specifically interested in the Web service area. The survey included questions to evaluate our proposed systematic model for maintaining the quality of a Web service artifact.

6.2.1 Experiment Settings

This section provides detailed information about the experiment settings in terms of the goal of the experiment, description of the experiment procedure, survey description, and description of survey participants.

- **Goal of the experiment**

  This research investigates and verifies a proposed (WDSQFD) methodology for applying Quality Function Deployment for Web Service description artifact.

- **Experiment Procedure**

  Participants were invited over the email and were asked to fill out a questionnaire using an online survey system. The expected time for filling out the survey is less than 15 minutes. Participants are expected to fill out the survey within one month from receiving the invitation over the email.

- **Survey Description**
The survey was built using an online system Qualtrics (Qualtrics, 2016). Qualtrics system was used to build the survey and distribute it to the participants over email. The survey consists of five questions to evaluate the proposed WSDQFD model and its steps. The online survey starts by displaying an informed consent to participate in the research study. The survey question will be displayed to the participants who agree to the consent. The survey questions consider of five questions as illustrated in Figure 20. The survey starts by asking the participants about their expertise in the domain of Web services. Then, the survey briefly explains the approach and asks the participants about his evaluation of the methodology in terms of the efficiency of the steps, the completeness of the steps, and the order on which these steps are performed.

Quality Function Deployment for Web Service Description Artifact Survey

1. Please explain your expertise in the domain of Web service.

We proposed a systematic analytical approach for creating a quality Web service description artifact. By Web service artifact we mean a Web service description that can be represented by an xml document which describes a Web service. The steps of this approach are illustrated in the following figure.

![Steps of Systematic Analytical Approach for Identifying Quality of Service Requirements of Web Service Artifact](image)

2. Do you think that these steps are efficient? Please explain why?
3. Do you think that one or more of these steps need to be removed or combined? Please explain why?
4. Do you think that there is any other step that needs to be added to the process? If yes what would it be?
5. Do you think that the order of these steps has to be changed? Please explain why?

Figure 20. WSDQFD Evaluation Survey Questions

68
Participants who were invited over the email were students and researchers who study or work on the Web service domain. Participants who are interested in the domain of Web service research who have been working in the Web service industry for more than one year are included in the survey. More than 250 email invitations were sent to people with an expected interest in the Web service domain. Five participants fully responded to the survey.

6.2.2 Experiment Results

This section reports the responses of the participants to the questions of the survey illustrated in Figure 20.

When asked if our model steps are efficient, most survey participants agreed that they are efficient because they are in an order that make the model efficient and that the steps are comprehensive. In addition one of the survey participants indicated that the efficiency can be enhanced by increasing the automation of these steps.

When the survey participants were asked if one of the steps should be removed or combined, most survey participants agreed that the steps are needed in the order to get consensus of variety of quality requirements from different perspective of different stakeholders. Some of the respondents indicated that the ranking and classifying steps can be clearly distinguished.

When asked if further steps should be added to the process, all survey participants agreed that there will be no need to add more steps to the proposed approach. One of the suggestions is that the model should involve an iterative feature to include new arising quality requirements for new versions of Web services.

When asked about the appropriateness of ordering the steps in the proposed model, most survey participants agreed that the steps are well ordered. One suggestion from the participants is that the classification step needs to come before the prioritizing step.
From the answers of the participants, it seems that our proposed model for maintaining the quality of a Web service description artifact is efficient. It can be enhanced by clearly distinguishing the difference between the later steps namely prioritizing and classifying Web service requirements. Additionally, indicating an iterative feature in the process will help ensure that new requirements to be included when a new quality requirement arises for an updated Web services.

6.3 Fully Automating the Steps of WSDQFD

The steps of WSDQFD illustrated in Section 6.1. can be fully automated by using natural language processing computerized methodologies. The collection of Web service quality requirements can be done by crawling the Internet to search for quality descriptions associated to Web services and legacy software programs. Also, media documents associated to Web services and software can be used to extract the quality attributes. Moreover, the text reviews of the Web services can also be used to extract these quality characteristics. The collected quality description can be further filtered using the language processing techniques to remove redundant quality attributes using language processing methodologies. In this step, the frequency of these attributes can also be counted. The prioritizing step can be used the quality attributes based on the frequency or based on the importance level of the source of the information. The classifying step can be done by using semantics and ontologies so the quality features can be clustered into specific categories. Fully automation the steps of WSDQFD could facilitate maintaining the quality of a Web service description artifact in short time. A limitation of using a fully automated system is that the methods would probably generate a number of errors because the methods can not accurately determine the semantics of the terms.
6.4 Summary

This section proposed a methodology for deploying quality function deployment to the domain of Web services. Specifically, we are concerned with characteristics of Web service quality description artifacts. A Web service description artifact has information about Web service so they would be published in the repository and retrieved later by requesters. Writing detailed and well-described Web service artifact makes it available and easily retrievable. The process of deployment quality model on the Web service artifact involved several steps including collecting, filtering, and prioritizing Web service quality requirements. The QFD can also be applied in the future to other aspects of Web service domain such as how to help providers build better services.
CHAPTER 7

Using Quality Function Deployment (QFD) to Design a Smart School Quality Factor Model: Integrating QFD into IoT

This chapter investigates the applicability of applying the concept of Quality Function Deployment (QFD) to the domain of Internet of Things and smart schools, and in fact, some parts of this chapter come from (Alnahdi & Melton, 2016). The Internet of Things (IoT) is the integration of things – all kinds of things – via the internet. This integration is done by having sensors on the things to collect data about the things, and then these data are shared via the Internet. This sharing enables the things to work together, and when the things work together, then the whole can be much greater than the sum of the individual things. An important question is how can all these things work together effectively?

This question is similar and maybe identical to the question which Quality Function Deployment (QFD) always asks: what is the voice of the customer regarding the important qualities of a product? When the engineers or producers of a product respect the voice of the customer, then the value, including the effectiveness, of the product is improved.

The immediate goal of this chapter is to show how QFD can be used in developing smart schools. The long-range goals are to make and support the claim that by integrating QFD into IoT, the value and usefulness of IoT will improve. Also, it should be noted that in IoT the voice of the customer is often the voice of a large and very diverse group. Thus, in IoT the way the voice of the customer is heard, understood, and applied may require a modification of QFD procedures.
An underlying theme of this chapter is although IoT is a significant step forward (in that when things, even seemingly unrelated things, are integrated and woven together, important progress can be made), more progress can be made when included in this integration is the voice of the customer. Thus, we are proposing that customers are included among the things which are integrated in IoT and that the voice or data of these customers become part of the shared data in IoT. This may also require a modified or new understanding of the voice of the customer.

In this chapter, we use “smart” to imply a part of IoT. Thus, for example, when we refer to a “smart school,” we mean that the school is part of IoT, i.e., the school is an IoT thing, and the school shares data with and interacts with other IoT things. By discussing smart schools, we can show how QFD can be integrated into IoT and also show in IoT that the diversity of the customers can be significant and that listening to this diverse voice may be important to the success of IoT.

If we consider, for example, a state supported school in the United States, the customers include students, faculty members, maintenance, and custodial staff for the school, the school administrators, and the tax payers who support the school. Not only are the persons in these different groups diverse – for example, they range in ages from teenagers to senior citizens, and their perspectives are vastly different. For example, there are students who want a school with an outstanding athletic program, administrators who want high quality teachers, taxpayers who want a low-budget school and custodians who want a building that is easy to clean and care for. Thus, when QFD is used in IoT, it must be able to listen to and interpret a rich and diverse voice. It may be hard to listen to and understand this voice, but it is rich, and the better it can be understood the better for IoT. In this chapter, we use “stakeholders” and “customers” interchangeably.
The rest of this chapter is organized as follows: Section 7.1 discusses previous work done in QFD and IoT. The process of building a quality factor model is provided in Section 7.2; the stages in building a quality factor model for smart schools are described in the subsections of Section 7.2. In Section 7.3 we discuss generalizing the use of QFD from a smart school to IoT in general. Section 7.4 reviews the main ideas for this chapter which includes integrating QFD into IoT and discusses possible future work.

7.1 State of art of QFD and IoT

(Chan & Wu, 2002) provided a comprehensive survey of QFD. Their survey covered 650 QFD publications. The history of QFD began in Japan in the late 1960s, moved to the U.S. in the 1980s, and then spread to other countries. The authors provided a list of QFD resources including QFD organizations, software, and online resources. In addition, they included a categorical analysis of QFD functional fields and methodological development. The core idea behind QFD is a way to transform customer requirements into technical requirements during stages of product development.

(Abdul-Rahman, Kwan, & Woods, 1999) distinguished between the QFD design approach and the QFD analytical approach. In addition, the authors differentiated between two types of customers: internal customers who work inside an organization which uses QFD and the external customers who receive and buy the product or the service. The external customers have the final word about the received service or product.

(Abdul-Rahman, Kwan, & Woods, 1999) also indicated that QFD consists of mathematical analyses that are based on functional relationships between matrices, and these analyses are used to quantify important qualities. The goal of the authors’ research was to study QFD applications and their applicability in the housing construction industry in Malaysia. The
authors used two types of survey questionnaires. The first survey was designed to get the developers’ point view. The second questionnaire was for those who would live in the constructed housing. The authors listed the important requirements or needs and then evaluated these based on importance. The authors also defined an improvement step in which decisions were made for each accomplished requirement. The authors used quality charts to determine priorities.

(Govers, 2001) distinguished between the what, how, and why of QFD. What are the customers’ requirements? How are these requirements translated into products? Why are certain requirements improved? Grovers claimed that the main goal of QFD is to find the relationship between design characteristics and customer requirements. In addition, the author classified problems with implementing QFD into three categories: (1) methodological problems including going into too many details, (2) organizational problems such as a lack of team dedication, and (3) product policy problems, for example, a lack of or inaccurate marketing information.

In (Hauser & Clausing, 1988) discussed House of Quality (HoQ) which uses matrices to enable producers and customers to work together to develop better products. A challenge in designing a product is that people have many viewpoints when it comes to quality. Before the industrial revolution, producer and customer were easily able to meet so the producer could translate the customer’s desires into a product. The steps the authors specified for using HoQ include determining the customers’ needs, called Customer Attributes (CAs), and prioritizing the important quality factors.

(Hauser, 1993) presented a company case study using HoQ. The author was fortunate, because it is uncommon for a company to be open and share its success stories with details. Hauser claimed that HoQ reduces design time by 40% and cost by 60% while maintaining
quality. Further, the author claims that recognizing the voice of the customer and voice of the engineer are essential when it comes to building quality products or services. The voice of customer may be gotten by interviewing 6 to 8 customers. However, customers may provide one hundred to four hundred needs or wants. Having lots of needs may make it difficult for a design or production team to work effectively. Therefore, an important step is to organize the needs as primary and secondary needs. The next step is to prioritize the needs by using QFD methods to estimate and balance the cost and the quality of the product for the good of the customer. Understanding customer perception is important to fill the gap between the company product and the best product. The voice of the engineer is also essential to identify important design attributes, to compare engineering procedures and metrics, and to develop the relationships and roof matrices.

(Kopetz, 2011) discussed IoT applications including energy savings, physical security and safety, industrial maintenance, and health monitoring systems. The author noted that the IoT vision includes giving each object a unique identifier (UID). Kopetz claimed that the “smart planet” is blossoming according to the IoT vision where everything has an intelligent identity.

(Su, Li, & Fu, 2011) demonstrated the structure classification of smart cities including construction layers including the perception layer, the network layer, and the application layer. The authors distinguished between a digital city and a smart city. A digital city is a city that includes systems such as a global positioning system (GPS), a geographic information system (GIS), and a remote sensor (RS). A smart city is a digital city that also includes intelligent sensors associated with other things, and it integrates the data from these sensors to create part of the IoT. Thus, a smart city could include, for example, a smart medical treatment center, a smart
tourism office, and smart transportation. A difficulty in maintaining a smart city includes managing the huge amounts of structured and unstructured data.

The authors in (Alnahdi, Melton, & Liu, 2016) used QFD to maintain the quality of Web service description artifacts. As the authors mentioned a Web service description is the main document that describes the Web service qualities and this document is used when searching for Web services by the users. The authors expected that concentrating on producing quality Web service descriptions would favorably impact fast and accurate Web service retrievals.

7.2 The process for Building a Quality Factor Model for A smart school

Determining the quality factors for building a smart school involves several steps which are shown in Figure 21. Detailed descriptions of the steps are provided in subsections 7.2.1 to 7.2.5.
When deciding on the qualities for a smart school, different stakeholders including decision makers and customers are involved in the process. Having a broad vision for different qualities from different stakeholders who are familiar with the domain knowledge and who own information about schooling systems, and experts in the field is essential when deciding on the specific qualities to be included in the quality model. In our model we would like to conform with the ISO standard (“ISO 16355-1,” 2015) where the priorities of stakeholders should be unbiased. In addition to those mentioned earlier in this chapter, there are parents, scholarship providers, financial engineers, IoT engineers, and quality engineers. Also different educational systems may have different stakeholders.

### 7.2.1 Deciding on the Stakeholders for Smart Schools Using IoT

When deciding on the qualities for a smart school, different stakeholders including decision makers and customers are involved in the process. Having a broad vision for different qualities from different stakeholders who are familiar with the domain knowledge and who own information about schooling systems, and experts in the field is essential when deciding on the specific qualities to be included in the quality model. In our model we would like to conform with the ISO standard (“ISO 16355-1,” 2015) where the priorities of stakeholders should be unbiased. In addition to those mentioned earlier in this chapter, there are parents, scholarship providers, financial engineers, IoT engineers, and quality engineers. Also different educational systems may have different stakeholders.

### 7.2.2 Determining Quality Factors from Stakeholders

The second step is collecting quality factors from stakeholders, including product manufacturer and service programmers. Quality factors are quality characteristics of services and
things in schools. The process of determining quality characteristics is an integral part of QFD. However, often in business and manufacturing situations, these stakeholders or customers form a homogeneous collection. The individual stakeholders will, of course, have different opinions, but they often have a similar general understanding of the business situation and the product being produced. Thus, often their differences are a matter of degrees and not fundamental differences. However, in our smart school example it is clear that some of these stakeholders have fundamentally different perspectives, opinions, and needs. Consider, for example, parents and financial engineers or scholarship providers and custodial staff members.

In this section, we give examples of IoT Web services and IoT components. Some classifications of Web services are as follows:

- **Security Web Services** may, for example, be attached to smoke detectors and fire detection devices and send appropriate messages to the fire department to help authorities handle hazardous situations.

- **Parking Space Availability Web Service** provides people with a way to know which parking spots are available when coming to events or on busy school days. It could also help people to direct car riders to the appropriate parking lots and spots during big events. The Parking Space Availability Web Service is a preferable service for students who study in crowded campuses. Also, it would be appropriate for people who have busy schedules.

- **Event Reminder Web Service** is to inform students who are in specific events nearby. This kind of service could help students who are studying to take a break so they can come back refreshed to school.
• **Promotion and Discount Web Service** is used to inform students of promotions and discounts at, for example, book stores and restaurants. A broadcast message can be sent to students who are subscribed to this service.

• **Schedule Change Notification Web Service** is used to inform students, faculty, and staff of updates of timing or location information about classes or meetings.

• **Weather Notification Web Service** is used to inform students, faculty, staff, and other employees of weather conditions such as the temperature, rain, and snow. Subscribers to this Web service will be prepared to dress appropriately for the weather. Also, they can be notified when there are severe weather conditions.

• **Work Backup Notification Web Service** is used to inform students, faculty, staff, and other employee who have available time to work and cover for an employee who has patient leave or travel necessity.

Quality factors of Web services are used to describe the quality features of Web services. Quality factors can be classified as: generic quality factors and domain specific quality factors. The generic quality factors are common for all Web services. Some of the generic qualities of Web services are specified in (Lee et al., 2003). On the other hand, domain specific quality factors are features that describe quality factors for specific Web services; in our case in this research, the domain is educational systems. Examples of quality factors for Web services that can be used in smart schools include the following.
• **Priority of a Smart School Web Service** is a generic quality feature that describes the importance of a Web service. Values for such quality features could be categorical values: low, medium, and high. For example, a security or weather notification Web service would get a high value. On the other hand, a promotion Web service might get a low importance value.

• **Availability of Web Service** is a generic networking quality feature that specifies how available a Web service is around the campus. Values associated to the availability could be determined as percentage. For example, a highly available Web service could be given a 98% while a Web service with low availability might have an 8% value.

• **Speed of Web Service** is a generic networking quality feature that describes how soon a Web service would be sent to a Web service subscriber. The values of this quality feature might have nominal values such as gold, silver, and bronze. For example, a Web service that is used for health or security should probably be gold. On the other hand, a Web service that is used for promotion or discounts could be classified as bronze.

• **Price of Web Service** is a generic quality feature that describes the price of a Web service. The pricing quality might vary based on the status of the subscriber. For example, students who improve their performance might get discounted prices for Web services.

• **Subscriber Classification of a Smart School Web Service** is a domain specific quality feature that specifies for which category of subscribers a Web service
is appropriate. The value of such quality features would be a nominal type; for example, a Web service might be for students, staff, or academics.

When it comes to smart schools, quality factors describe both the software components represented as Web services and the hardware components also known as things in the domain of IoT. Earlier in this section, we listed some of the Web services to be included in smart schools and the quality factors of these Web services. The quality factors of the things include the following:

- **Durability** is to describe strengths of the material of devices and wires. Educational campuses that are located in locations where there are severe weather conditions should have high durability measures of their hardware components.

- **Sustainability** is to describe how long the hardware components will continue to work.

- **Suitability** is to describe how easily hardware or a device fits in the school system. For example, schools that have small storage rooms would prefer to purchase small computers, and server machines.

- **Usability** is to describe the level of ease of using hardware components.

### 7.2.3 Filtering and Prioritizing Quality Factors

As mentioned in the Literature Review Section, a group of customers may identify several needs or wants, and having many needs may stifle the productivity of the design and development teams. In this smart school scenario, we have multiple groups of customers, and we could end up with thousands of potential needs. Filtering and prioritizing those needs is essential for resources saving and producing enhanced customer oriented quality model. Filtering and
prioritizing the quality factors given by step number 7.2.2 is based on several norms; Prioritizing needs can be achieved by the ranking score of those needs. Needs that have been frequently suggested by several stakeholders would have high priority. Needs that are provided by highly ranked stakeholders would have high ranking. Also, customer needs that can be satisfied by resources that are already available in the school environment would have high ranking. Moreover, the level of experience of stakeholders is essential in deciding the ranking of the needs, the more experienced the stakeholder the higher the ranking of the provided specification.

7.2.4 Building a Quality Factor Model

In this step, a matrix with two or more dimensions is created. This matrix allows us to visualize the important quality factors, and thus, we are better able to appropriately use the quality factors in producing the best quality products. Visualizing the quality factor artefact is the process of describing the quality factor specifications as a graph to help decision makers make their decision. In section 7.1, we have mentioned how using the House of Quality (Hauser & Clauing, 1988) is helpful as a tool to decide quality factors. An example of a quality matrix that represents Web services and quality factors of the smart school model can be seen in Figure 22.
Figure 22. Quality matrix for smart school Web services
7.2.5 Normalizing Quality Factor Model

Normalizing the Quality Factor Solution is the final step of the process of determining quality factors which yields the final candidate quality factors. After building a quality factor model, quality engineers will be able to make the final quality factor decisions for the system. Quality engineers are responsible to decide the final selection of the quality factor solution. Depending on the visualized artefact, they would be able to decide what the final Web services and components to be included in the system and what quality factors are to be selected for those Web services and components.

7.3 QFD AND IOT

IoT is an important development which builds on and uses the internet and also makes use of Semantic Web ideas and recent progress in big data. A key component of IoT is a belief that by integrating and analyzing data about IoT things the potential and effective uses of these things will significantly increase. Of course, this belief has become more credible as our ability to handle big data has improved. A premise of this chapter is that the effectiveness and uses of IoT things would also improve if we added a voice of the customer to IoT itself. This would mean that IoT would not only have data from the things already in the IoT, but IoT would have another source of data, i.e., the voice of customers, and another thing, i.e., customers. These ideas and possibilities raise interesting questions.

How would customer/stakeholder things be organized? How would the voice, i.e., data, of the customer be interpreted by the other things, i.e., how would the voice of the customer be integrated with the data from the other things?
• How would customer/stakeholder things be organized?

Customers/stakeholders things can be organized based on the classifications of those stakeholders. One way of classifying stakeholders is a binary classification such as {receiver of the service and provider of the service}. Another way of classifying stakeholders is based on hierarchal classification such as status classification {Students, Faculty, and Administrative, Parent, and/or Scholarship provider}.

• How would the voice, i.e., data, of the customer be interpreted by the other things, i.e., how would the voice of the customer be integrated with the data from the other things?

The voice of the customer provides preliminary and mostly insightful opinions about the functionality and quality of services and equipment in the smart school. However the log data and the feedback data from users would give an indication about the importance of those services.

• Could the voice of the customer play an interpretive role with respect to the data of the other things? Could the other things have an interpretive role with respect to the voice of the customer?

The voice of the customer is essential when analyzing received data from other things. For example, service usage data, log data, and feedback data can be shown to the stakeholders of the system in a statistical format to decide which services and products are actually useful to the consumer. Also, statistics feedback information can be used to find out which services and/or products are frequently used. Moreover, a failure reporting
system would be beneficial to signal non-functional devices or services. This kind of integration would help to improve the quality of the overall system.

- Should IoT include a QFD thing which could employ QFD techniques in IoT so that QFD methods could continually be applied to IoT as a continually improving part of IoT?

A proposed QFD thing is to be used for handling and monitoring QFD functionalities. The QFD thing can be composed of several things such as services for collecting feedback from users for example, statistical services for data visualization and interpretation, services for analyzing usage data, and feedback data, services for interpreting data, and decision support services that is used by the QFD engineers for adjusting the system by enhancing, adding, or reducing services.

In the State of Art of QFD and IoT Section, Section 7.1, we mentioned that before the industrial revolution, producers and customers could easily meet so that the producers would know what the customers wanted (Hauser & Clausing, 1988). We may be coming full circle in that with effective communication of data in IoT, producers and customers may be able almost instantaneously to know what the other wants or expects. In fact, using QFD methods, the producers and customers may be able to interact so that both the customers’ understanding of what the producers do and the producers’ understand of what the customers want and need might both improve, and even more interestingly, might converge so that they might work together for the benefit of both. However, to do this IoT may have to have QFD built into it.
7.4 Summary

In this chapter, we use QFD and IoT to build quality factor models for smart schools. We propose a procedure to build quality factor models. This procedure begins with identifying customers who propose quality factors. These quality factors are then filtered and prioritized so that a quality factor model can be produced. Then the quality factor model is normalized.

After using QFD to design quality factor models for smart schools, we pose the question of incorporating QFD into IoT. We make the case that by incorporating QFD into IoT the quality and effectiveness of IoT could be significantly improved. We raise some interesting questions which can be addressed when trying to determine the best ways to incorporate QFD into IoT. Also, we addressed how these issues can be handled.
CHAPTER 8

Conclusion

This dissertation provides a quantitative approach for retrieving and ranking Web services based on qualities. Retrieving Web services based on qualities is essential when the user must choose from functionally equivalent Web services. It is important to have a Web service information retrieval system that provides customizable retrieval criteria that support the user’s quality requirements.

The methodology of our retrieval approach is based on measuring the similarity between quality characteristics of a query and an offer. The similarity measurement process is based on using algorithms that estimate the similarity between the query and available offers. The smaller the distance between a query and an offer, the more the relevancy between that query and the offer.

Quality Function Deployment (QFD) and House of Quality concepts (HoQ) from business domain can be applied to the domain of Web service retrieval. This research provided a vision on how QFD can be applied to the domain of Web service retrieval and quality. Specifically, it focuses on how to apply the quality on the Web service description artifact because it is the essential component for the retrieval process. We provided a procedural process on how to build a Web service description artifact that adheres to quality characteristics. Having an artifact that describes the Web service well enough will enable that Web service to be accurately retrieved.
In addition, this dissertation provided a model for applying QFD to the domain of the educational system. Incorporating Internet of Things (IoT) to educational systems will provide smart schools. Smart schools will incorporate both hardware (i.e. things) and software components (Web services). Identifying what components to include in the system and what quality parameters are there is a process that can be done by using the proposed quality model. In addition, the collected data from the smart school things is essential and considered as feedback components to further enhance the smart school educational system.
CHAPTER 9

Future Research Directions

In this dissertation, we studied quality of Web service retrieval. Also, we studied how quality function deployment can be applied when it comes to the domain of Web services. Future research directions that can extend our work would be studying the quality function deployment and how can they applied to the domain of creating Web services. Are these factors similar or different? How would someone compose these factors together to build a quality retrieval system model?

Another research direction could be measuring procedures for the quality of a Web service quality data. For example, if there is a party that provides Web service Quality data, how will a researcher know if these data maintain good quality for computing model training? For example, does the dataset have ample attributes that describes Web service quality characteristics? Does the dataset have enough records to be used? Does the dataset has complete values or it is sparse? Does the dataset have been used by other researchers for training computational model? Having metric suites for Web service dataset will enable researchers who work on Web service domain to build computational models by comparing different available Web service quality dataset using metric suites.

Web service recommendation is another promising research area. When somebody selects a Web service, a Web service recommendation system might provide a
similar or complimentary Web services that are close in quality to the appropriate Web
service. Moreover, a Web service recommender system might be built on studying the
historical behavior of Web service users. So, for a Web service user who selected a Web
service WSA, the system might recommend a Web service WSB to the user because it
has been learned from the system that other users have selected WSB after selecting
WSA. In addition, it would be interesting to have a quality model that evaluates Web
service recommendation systems and classifies them based on their methodology,
efficiency, uses, and popularity.

Graphs are data structures that have proven its success in solving several
problems in the domain of Computer Science. In the domain of Web service retrieval,
graphs has shown its success when it comes to the functional aspect (Ma & Lee, 2012)
have proposed a graph-based approach for Web service matchmaking based on functional
requirements. The authors measured the similarity between the request graph and service
graphs. The request graph is built from a form that the user fills in by entering the desired
functional requirements. On the other hand, the service graphs are built by parsing the
WSDL descriptions of Web services. Both graphs consist of three portions: input, output,
and operations. Graph nodes are classified into three types: connector nodes, data type
nodes, and keyword nodes. The authors provided a set of similarity definitions (crisp and
fuzzy). The authors dealt with functional aspects of Web service. It would be interesting
to see if graphs could be used in the domain of Web services when it comes to Web
service retrieval based on quality.
REFERENCES


Christensen, E., Curbera, F., Meredith, G., Weerawarana, S., & others. (2001). *Web services description language (WSDL) 1.1.*


99


