Quality of Service Assessment over Multiple Attributes

Hmood Zafer Al-Dossari

A thesis submitted in partial fulfilment
of the requirement for the degree of Doctor of Philosophy

School of Computer Science & Informatics
Cardiff University

June 2011

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Abstract

The development of the Internet and World Wide Web have led to many services being offered electronically. When there is sufficient demand from consumers for a certain service, multiple providers may exist, each offering identical service functionality but with varying qualities. It is desirable therefore that we are able to assess the quality of a service (QoS), so that service consumers can be given additional guidance in selecting their preferred services. Various methods have been proposed to assess QoS using the data collected by monitoring tools, but they do not deal with multiple QoS attributes adequately. Typically these methods assume that the quality of a service may be assessed by first assessing the quality level delivered by each of its attributes individually, and then aggregating these in some way to give an overall verdict for the service. These methods, however, do not consider interaction among the multiple attributes of a service when some packaging of qualities exist (i.e. multiple levels of quality over multiple attributes for the same service).

In this thesis, we propose a method that can give a better prediction in assessing QoS over multiple attributes, especially when the qualities of these attributes are monitored asynchronously. We do so by assessing QoS attributes collectively rather than individually and employ a k nearest neighbour based technique to deal with asynchronous data. To quantify the confidence of a QoS assessment, we present a probabilistic model that integrates two reliability measures: the number of QoS data items used in the assessment and the variation of data in this dataset. Our empirical evaluation shows that the new method is able to give a better prediction over multiple attributes, and thus provides better guidance for consumers in selecting their preferred services than the

existing methods do.

To my parents,

my wife,

and my beloved children Bassam, Ghaday and Kadi

Acknowledgements

First of all I thank and praise Allah (God) Almighty for all his favours on me and for providing me with faith, patience and commitment to complete this research.

I am privileged to have Dr. Jianhua Shao as my supervisor. The high standard of his research has always been an inspiration and a goal to me. Dr. Shao always pushed me to bring the best out of me, had confidence in me, and worked very hard to transform me from a struggling graduate student to a researcher that is able to investigate, identify, and address real research problems. I am deeply grateful to him for his consistent encouragement, invaluable guidance and strong support during the course of this study. What I have accomplished so far would not be possible without Dr. Shao's guidance and support. His thoughtful advice and constant sharing knowledge and experience will not be forgotten.

I would like to thank Professor Alun Preece, my second supervisor, for his valuable feedback of joints papers. I would also like to thank my annual reviewers panel, especially Professor Steve Hurley, for their constructive comments. My thanks also goes to the member of the school for their help, especially Mrs. Helen Williams and Dr. Pamela Munn for their help on administrative issues, and Mr. Robert Evans and Dr. Rob Davies for their technical assistance.

I wish to thank my colleagues, Dr. Fahad Al-Wasil, Dr. Badr Aldaihani, Dr. Ahmed Alqaoud, Waleed Alnuwaiser, Yahya Ibrahim, Ahmed Alazzawi, Ehab ElGindy and Abdul Hamid Elwaer, for their friendship and help. Special thanks must goes to Sultan

Alyhaya with whom I have worked closely on approximate regularities. I have enjoyed our numerous discussions on the subject and many other topics. I would like also to thank many friends I have, the Saudi students in Cardiff, for providing me with plenty of good time and distractions from work. Having them in Cardiff made me realise how fortunate I am to be with such good people. I acknowledge, with grateful thanks, the Saudi government represented by King Saud University for sponsoring me throughout the research period.

My special admiration and gratitude to my parents, who above all people have made me the person I am today, for their prayers, care, love, patience during my absence for much of the past years, and teaching me never to give up. My sincere gratitude also goes to my sisters, brothers, aunts and uncles whose their prayers and encouragement have always enabled me to perform to the best of my abilities. Special gratitude must goes to my friends in Saudi Arabia, especially to my uncle Shaya Alshkarah and my closest friend Faleh Alwadani for taking care of my personal issues in Saudi Arabia and keeping in touch at all times.

Last, but certainly not least, I am indebted to my wife Haya for her endurance, patience, sacrifices and unconditional support during the period of my PhD. Without her love, selfless decisions, practical advice and tireless encouragement this research would have been impossible. Finally, my deeply indebted to my beloved children, Bassam, Ghaday, and Kadi who have given me happiness during the difficult period of my study.

Contents

List of Figures

List of Tables

Acronym

AA	Averaging All	53
KNN	K Nearest Nieghbour	87
$LOCF$	Last Observation Carried Forward	84
MCV	Most Common Value	83
$MQSM$	Multiple Quality Space Mapping	54
PDF	Probability Density Function	33
PCC	Pearson Correlation Coefficient	30
QoS	Quality of Service	1
$QoS - IC$	QoS Information and Computation	23
RI	Random Imputation	83
SE	Synchronous Extension	58
$TRAVOS$	Trust and Reputation for Agent-based Virtual OrganisationS	34
$UDDI$	Universal Description Discovery and Integration	21
UX	UDDI eXtension	27
$WSAF$	Web Services Agent Framework	25
$WSMO$	Web Services Modelling Ontology	24
WSP	Web Service Procurement	24
$WS - QoS$	Web Service QoS	23
$WSRec$	Web Service Recommendation	30

Chapter 1

Introduction

The development of the Internet and World Wide Web have led to many services being offered electronically. When there is sufficient demand from consumers for a certain service, multiple providers may exist, each offering identical service functionality. In this instance, the consumer must make a decision as to which provider to approach to obtain the service. In addition to functional requirements, a consumer's decision will be based upon requirements that are related to *how* a provider delivers the service (i.e. non-functional requirements). Non-functional requirements are also referred to as the quality of service (QoS) and often play a significant role in a consumer's decision-making process [10].

It is well established that people are goal seeking [32] and their goals can be achieved by combining a variety of resources to complete a task. However, the required resources may not be immediately available, so it is necessary for people to interact with providers who are willing to supply the resources they require to achieve their needs. If software components are to effectively support consumers in deciding which provider to select, they must be able to represent these factors that are important to consumers when choosing a provider, including those concerned with QoS.

Research suggests that consumers go through a five-stage process when making decisions [133] about which service to use. The five-stages are illustrated in Figure 1.1.

1. **Need Recognition.** The consumer's functional and non-function requirements are determined in this stage.

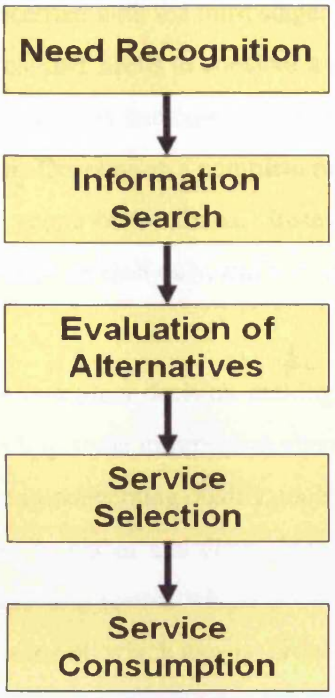


Figure 1.1: The consumer's decision-making process [133]

2. **Information Search.** The consumer attempts to discover all possible providers who claim to be able to offer the required capabilities identified in the first step. Once identified, the consumer will gather as much additional information about each of the providers as possible.

3. **Evaluation of Alternatives.** The consumer will evaluate each of the providers against the needs identified in the first stage based on the evidence gathered in the second stage.

4. **Service Selection.** The consumer will select the preferred provider based on the outcome of the third stage.

5. **Service Consumption.** The consumer will invoke and use the selected service.

This thesis is primarily concerned with the third stage: evaluation of alternatives. This is a complex issue, as a consumer needs to consider a range of factors in weighing up alternatives, for example, the quality and cost of a service, the reputation of a provider, third party recommendation. Developing a complete model and solution to address all these issues is beyond the scope of the thesis. Instead, this work will focus on the evaluation of alternatives based on QoS only, and for other issues, the reader is referred to [38, 64] for details.

Incorporating QoS into the consumer decision-making process is not trivial and it requires more than just including some information about quality in service description. For effective decision-making concerning quality, tools must be deployed within a service provision environment to gather and disseminate information regarding the behaviour of service providers in a format which is useful to consumers. This can be achieved through QoS assessment, which may be broadly described as a function to determine, using historical service provision data, the likely quality that a consumer may get from a service provider.

Being able to assess the quality of a service is desirable. This is because when multiple providers offer functionally identical services, service consumers can be given additional guidance in selecting their preferred services. The major contribution of this thesis is the development of a method that gives a better prediction in assessing QoS over multiple attributes, especially when the quality of these attributes is asynchronously monitored. This helps consumers to make appropriate decisions to select a service that best meets their needs.

The rest of this chapter is organised as follows. Section 1.1 presents a specific scenario to help illustrate the challenges associated with the task of QoS assessment. This scenario is used throughout this thesis to explain the shortcomings of existing approaches to QoS assessment, and how the method proposed in this thesis addresses these limitations. Section 1.2 details research objective and methodology to address the problem. Section 1.3 highlights the contributions of the research, while Section 1.4 describes the

organisation of the thesis.

1.1 Motivating Scenario

To help illustrate QoS assessment and highlight the challenging issues involved in assessing QoS over multiple attributes, we introduce a specific scenario from the travel industry in this section.

Alice is a tourism services provider. She provides travel services such as airline bookings, hotel reservations and car rentals. Motivated by recent developments in technology, she has decided to move her business from physical to electronic services. So, a virtual travel agent (VTA) has been implemented to provide personalised and customised assistance and automation to Alice's customers.

A web hosting service is required to deploy the VTA and to make it available to customers. However, down time of a web server means that hosted e-commerce sites lose business and excessive delays turn into dissatisfied customers that lose patience [10, 13]. The performance and high-availability of the potential hosting web service are therefore critical to the continuity of operations of the VTA and the success of Alice's business, especially with a large number of customers. Accordingly, Alice needs a good hosting service for her business.

Now, assume that Alice wishes to find a web hosting service that can satisfy a certain level of QoS. Suppose that Alice's preferences are expressed as follows: `access delay` = 200 milliseconds (the round trip time between sending a request and receiving a response) and `throughput` = 800 requests per second (the number of requests served per second).

Typically, many service providers offer web hosting services at different levels. Suppose that based on service advertisements, Alice has found four web hosting service providers ($S1$-$S4$), as shown in Figure 1.2, who offer the requested level of service

(`access delay`=200 milliseconds and `throughput`=800 requests/second). So, Alice will need to make a decision about which service provider to select.

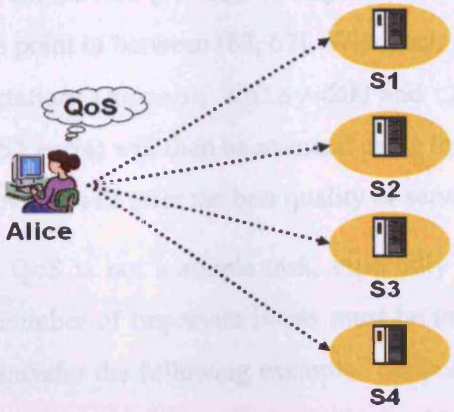


Figure 1.2: Available Web Hosting Service Providers

One way is for Alice to choose a service provider randomly. However, this may not give Alice the best provider available. This is because service providers may not be entirely trustworthy. In other words, what they claim may not be what a service requester will get. For example, $S3$ may advertise itself as offering short `access delay` (e.g. around 200 milliseconds), but time will perhaps tell that this claim is not genuine and web sites hosted by $S3$ may not always be accessible (i.e. `access delay` may be longer than what has been advertised). Also, in a dynamic service provision environment, the quality of service may not be stable. For example, $S4$ may claim that it can handle 800 requests per second, but in the actual service provision it may in fact drop below that level. In such an uncertain and dynamic environment, Alice would welcome an accurate and objective verdict on the candidate service providers in terms of whether they will deliver what she expects.

QoS assessment can help Alice in the above scenario. The idea underlying QoS assessment is to collect historical service provision data for the service concerned and use this to determine the likely service level that a consumer may get from a service provider.

This assessment is carried out by an independent, trusted party within the service provision environment, with QoS data collected using monitoring tools residing locally on the server providing the service [5, 152], locally on the consumer side using the service [22], or at some point in between [83, 67]. With such a unit, Alice should be able to specify her expectations (`access delay`=200 and `throughput`=800) and the candidate services ($S1 - S4$) will then be assessed using their historical service data to determine which provider will offer the best quality of service w.r.t. her expectations.

However, assessing QoS is not a simple task, especially over multiple attributes as in Alice's case. A number of important issues must be taken into account. To illustrate these issues, consider the following example. Suppose that some historical data about the performance of $S1$ w.r.t. `access delay` and `throughput` has been collected as shown in Table 1.1. Each tuple in Table 1.1 represents a single service provision instance that has been observed by some monitoring tool. TID is the identifier of the instance, and the other two columns are the monitored `access delay` and `throughput` delivered by $S1$ for that instance.

Table 1.1: Monitored QoS Data for $S1$

TID	*Access Delay (milliseconds)*	*Throughput (requests/second)*
t001	570	400
t002	200	815
t003	630	380
t004	155	780
t005	600	450
t006	230	850

To assess the quality for $S1$ we may average the observed data to obtain $Avg(accessdelay) = 397.5$ and $Avg(throughput) = 612.5$. This type of calculation of QoS is meaningful if we assume that every level of quality is deliverable by $S1$. When this is not the case, such methods can generate misleading verdicts. To see this, we re-arrange the order of

tuples (service provision instances) in Table 1.1 to give Table 1.2.

Table 1.2: Re-arranged QoS Data for $S1$

TID	*Access Delay (milliseconds)*	*Throughput (requests/second)*
t001	570	400
t003	630	380
t005	600	450
t002	200	815
t004	155	780
t006	230	850

Now we can observe two groups of service provision instances in the table: one group with `access delay` offered at around 600 and `throughput` at around 400, and the other with `access delay` delivered at around 200 and `throughput` at 800. If we assume that this grouping is not accidental, and $S1$ in fact offers only these two possible quality levels, perhaps as a result of some resource management requirement [127], then our prediction of $QoS_{S1}(accessdelay) = 397.5$ and $QoS_{S1}(throughput) = 612.5$ is clearly misleading as the predicted quality is not possible to obtain in practice.

One possible solution to this problem is to assess QoS per user expectation [127]. To illustrate how this method works, we assume that user expectations for services are also collected, and the expanded data is shown in Table 1.3. If Alice is seeking a web hosting service that can deliver `access delay`=200 and `throughput`=800, then we can select the data from Table 1.3 w.r.t. this requirement, and then aggregate the selected data to assess $S1$ as before. For instance, we can select these instances that satisfied $|expected(accessdelay) - 200| \leq \delta$ and $|expected(throughput) - 800| \leq \delta$, and if we let $\delta = 100$, then we will have $Avg(accessdelay) = 195$ and $Avg(throughput) = 815$ (i.e. averaging over `t002`, `t004` and `t006`), which is a more accurate assessment of what $S1$ is capable of delivering.

Table 1.3: Historical QoS Data for $S1$ with Expectations

TID	*Access Delay* $\langle expected, delivered \rangle$	*Throughput* $\langle expeceted, delivered \rangle$
t001	$\langle 650, 570 \rangle$	$\langle 500, 400 \rangle$
t002	$\langle 240, 200 \rangle$	$\langle 850, 815 \rangle$
t003	$\langle 800, 630 \rangle$	$\langle 400, 380 \rangle$
t004	$\langle 120, 155 \rangle$	$\langle 770, 780 \rangle$
t005	$\langle 500, 600 \rangle$	$\langle 520, 450 \rangle$
t006	$\langle 270, 230 \rangle$	$\langle 880, 850 \rangle$

However, this approach implicitly assumes that QoS data over multiple attributes is synchronously collected. This is not realistic as, in practice, multiple QoS attributes are more likely to be monitored independently and at different rates [148, 66, 130]. When QoS data across multiple attributes is not synchronously collected, the approach suggested in [127] could have substantially less data to use. This may affect the confidence of assessment. To illustrate this, assume that the data given in Table 1.3 is in fact collected as in Table 1.4. Clearly, in this case, only `t008` will be selected by the expectation based method in assessment. While this still gives a correct assessment, intuitively our confidence with this assessment will be low, as only a very small fraction of the data is actually used in assessment.

In this thesis, we will discuss how asynchronously collected data may affect QoS assessment over multiple attributes. We will then describe how our proposed method in this thesis may help addressing this issue to produce more accurate and confident assessment.

Table 1.4: Asynchronously Collected QoS Data for $S1$

TID	*Access Delay* $\langle expected, delivered \rangle$	*Throughput* $\langle expeceted, delivered \rangle$
t001	$\langle 650, 570 \rangle$	
t002		$\langle 500, 400 \rangle$
t003	$\langle 240, 200 \rangle$	$\langle 850, 815 \rangle$
t004	$\langle 800, 630 \rangle$	
t005		$\langle 400, 380 \rangle$
t006	$\langle 120, 155 \rangle$	
t007		$\langle 770, 780 \rangle$
t008	$\langle 500, 600 \rangle$	$\langle 520, 450 \rangle$
t009	$\langle 270, 230 \rangle$	
t010		$\langle 880, 850 \rangle$

1.2 Research Objective and Methodology

The aim of this thesis is to address the issues arising from assessing QoS over multiple attributes. More specifically, our aim is to *develop a method that takes both accuracy and confidence into account when assessing QoS over multiple attributes, assuming that the historical QoS data for the attributes are asynchronously collected.* This overall aim leads to several research questions:

1. How good are existing QoS assessment methods in dealing with multiple attributes?
2. How to establish confidence for QoS assessment?
3. How to handle asynchronous QoS data in assessing QoS over multiple attributes?

Together, these questions form our research hypothesis. In addressing these research questions, we have followed the following steps:

Step 1. To address the first research question, we define a set of requirements that should be satisfied by an effective QoS assessment method, and analyse and compare existing QoS assessment methods against these requirements. Our investigation covers a range of fields, including trust and reputation in multi-agent systems [60] and QoS management in service provision environments [102]. The reviewed QoS assessment methods are classified into different groups based on their characteristics. The shortcoming of these methods are identified in the context of this research.

Step 2. The second research question is tackled by exploring how confidence is established in other fields of studies, such as in trust and reputation systems. A set of factors that may affect the confidence of QoS assessment are examined and then taken into account in developing a confidence model for QoS assessment.

Step 3. To address the third research question, we treat asynchronous values as missing values and attempt to predict such missing values as commonly exercised in data mining [39]. We evaluate and compare a number of imputation methods based on a set of rules produced by Sande in [121], and then follow one of them to handle asynchronous data. Our investigation ranges from simple and straightforward imputation methods to more advanced and conceptually more complex methods based on machine learning.

Step 4. To validate the research results, we perform several experiments representing some real world scenarios. We implement an environment which allows the behaviours of providers and consumers to be modelled, and interactions between service consumers and providers to be simulated. We compare a number of assessment methods in our experiments.

1.3 Research Contributions

This research makes the following contributions:

- **A conceptual model for QoS assessment.** We propose an abstract model for understanding QoS assessment. We use this model to describe and contrast approaches to quality assessment, and to guide the development of a specific QoS assessment method described in this thesis. This model is generic, and hence can be used as a guideline for any further enhancement to QoS assessment approaches.

- **A probabilistic model to quantify confidence in QoS Assessment.** We present a probabilistic model to quantify confidence in QoS assessment. We do so by integrating two reliability measures: the number of QoS data items used in assessment and the variation of data in this dataset. We show that our confidence model can help consumers to select services based on their expectations more effectively.

- **Handling asynchronous QoS data.** We treat asynchronous data as a dataset containing "missing" values and we attempt to predict such missing values. To do so, we evaluate a range of data imputation approaches to predicting missing data, and then suggest the use of a k Nearest Neighbour based technique to predict asynchronous values. We show that, by handling asynchronous data suitably, our proposed QoS assessment method can improve the confidence of QoS assessment over multiple, asynchronously monitored attributes.

The proposals from this research are evaluated using a software simulation environment. The simulation environment is used to test and compare approaches to QoS assessment, supporting fine-grained control over scenario parameters and evaluation of QoS assessment methods' performance. Our experiments show that the proposed method results in a more accurate and reliable QoS assessment.

1.4 Thesis Structure

The rest of the thesis is organised as follows:

- **Chapter 2** presents a review of existing methods for assessing QoS in different fields, including trust and reputation in multi-agent systems and QoS in service provision environments. The reviewed methods are identified, and their characteristics are described. In the light of these characteristics, we identify several significant limitations of existing QoS assessment methods which we aim to address in the subsequent chapters.
- **Chapter 3** provides a conceptual model for describing issues relevant to QoS assessment and draws out a definition of quality that will be adopted in this thesis. Through this conceptual model, we discuss how effective QoS assessment methods may be developed.
- **Chapter 4** presents a probabilistic model to quantify confidence in QoS assessment. Particular attention is paid to determining the contribution of each component in this model to its overall performance. We show how the proposed confidence model can be used to help consumers to select services based on their expectations more effectively.
- **Chapter 5** describes our proposed solution to handle asynchronous QoS data when assessing QoS over multiple attributes. How to integrate the proposed solution into existing QoS assessment methods to enhance their performance is explained.
- **Chapter 6** presents a set of criteria for the evaluation of QoS assessment methods, and conducts a set of experiments to demonstrate the performance of our proposed method against existing methods under a range of realistic and informative scenarios.
- **Chapter 7** concludes the thesis and outlines the directions for future work.

Chapter 2

Related Work

This chapter reviews the state-of-the-art QoS assessment methods. We describe each method and analyse its strengths and weaknesses. The chapter is organised as follows. It starts with different views of quality used by the existing quality assessment methods in Section 2.1. Section 2.2 introduces a quality cycle to facilitate the discussion on different aspects of QoS assessment, followed by highlighting several challenges associated with QoS assessment and a set of requirements that must be considered in developing a sound QoS assessment method in Section 2.3. Section 2.4 analyses and compares the reviewed QoS assessment methods to our work, evaluating how they fulfill the requirements described in Section 2.3 and identifying the research issues and challenges to be addressed in the subsequent chapters. Finally, Section 2.5 summarises the findings of this chapter.

2.1 Quality Paradigms

There is no consensus in the literature on exactly what quality is, although its pervasive importance has been recognised. Broadly, there are two views of quality: quality as reputation and quality as conformance [27].

Quality can be linked to reputation which can be defined as the perception that a service provider builds over time about its intentions and norms [96]. For example, an Internet service provider may be considered to deliver good quality if it is consistently fast

and stable over many years. There exists a large body of studies on quality as reputation [113, 157, 153, 60]. In these studies, it is typical that the observations on service providers' behaviour are obtained from service consumers over time as ratings. That is, after each service provision, service consumers are asked to indicate their perception of how well the services have been delivered. These ratings are usually expressed on a scale, for example from 1 to 5 where 1 indicates bad quality and 5 excellent quality, as shown in Figure 2.1. The collected ratings can then be computed and analysed to indicate the quality of service.

Table 2.1: QoS verdict by ratings

1	2	3	4	5
Bad	*Poor*	*Fair*	*Good*	*Excellent*

Quality as conformance, on the other hand, links quality to the degree to which a service provider meets (or conforms) to a particular 'ideal' level. This ideal may be defined by individual service consumers as their requirements or expectations. In contrast to the reputation view of quality which is merely a user perception, conformance view of quality indicates the success or failure of service providers in terms of delivering consumers' expectations. Some models in the literature adopt the conformance view of quality [64, 126, 127, 149]. In these models, QoS is calculated as the difference between the level of service delivered by service providers and the expectation of service consumers. That is, QoS is measured in terms of how well a delivered service meets a user's expectation. Since it is possible for different consumers to perceive the same delivered level of service differently, the subjectivity between different consumers is explicitly recognised in this view of quality.

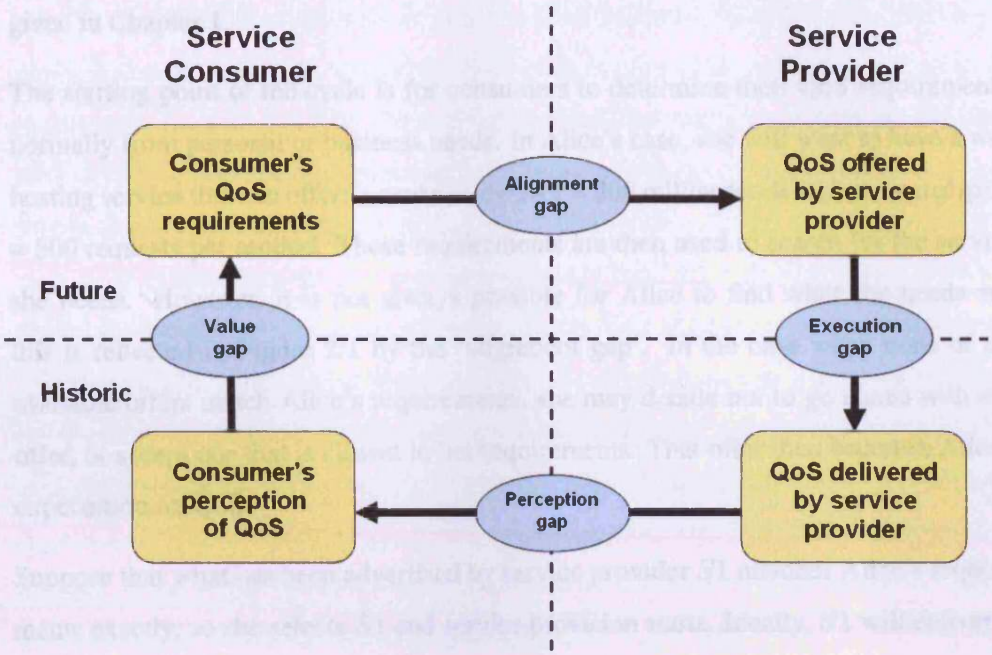


Figure 2.1: Quality Cycle adapted from [101]

2.2 Quality Cycle

To discuss the issues related to QoS assessment, we introduce a quality cycle adapted from [101] as shown in Figure 2.1. The two principal parties in the cycle are service providers and consumers. In an open market, providers publish their services and consumers request services, and their interactions over time form a cycle. That is, consumers search through advertised services to find the most suitable one, both functionally and QoS-wise, and then bind to the chosen service provider for service consumption. During service provision delivered quality is monitored, captured and stored. This information is then utilised in future by the consumers to improve their selection of a particular provider who can meet their requirements better. Ideally, consumers should be able to fulfill their needs through such interactions. However, there are several 'gaps' in this cycle, which mean that consumers may not always get what they expect in practice. In the following, we explain these gaps using the Alice example

given in Chapter 1.

The starting point of the cycle is for consumers to determine their QoS requirements, normally from personal or business needs. In Alice's case, she will want to have a web hosting service that can offer: `access delay` = 200 milliseconds and `throughput` = 800 requests per second. These requirements are then used to search for the service she needs. However, it is not always possible for Alice to find what she needs and this is reflected in Figure 2.1 by the 'alignment gap'. In the case when none of the available offers match Alice's requirements, she may decide not to go ahead with any offer, or accept one that is closest to her requirements. That offer then becomes Alice's expectation on QoS.

Suppose that what has been advertised by service provider $S1$ matches Alice's requirements exactly, so she selects $S1$ and service provision starts. Ideally, $S1$ will deliver to Alice what has been advertised or required. In practice, however, this may not always be the case since the delivered level of service by $S1$ to Alice may deviate from the advertised one, either intentionally or unintentionally [60, 89]. For example, the provider of $S1$ may over advertise his capability in order to attract consumers and the advertised level may not be achieved in practice. Moreover, even though providers are true to what they have advertised, their delivered level may still deviate from advertisements due to some factors that are beyond their control. For example, as a result of some effective recommendations, a large number of consumers use $S1$ making the resource of its provider overloaded and consequently it delivered a level of quality to the consumers lower than what was advertised. The variation between what was advertised and delivered is the 'execution gap' in Figure 2.1. In an open service provision environment, service monitoring normally takes place between service providers and consumers to gauge if the providers have delivered what they have advertised [4, 84, 119]. Such monitoring activities result in a collection of QoS data that can be used in future to help consumers in assessing providers' offers.

Unfortunately, even if the provider of $S1$ has delivered what he advertised, it is still

possible for different consumers to perceive the QoS from $S1$ differently. This is because, different consumers have different quality requirements, and their perception of quality may be influenced by their requirements. The study conducted by Zeithaml et al. [158] showed that consumers' perception of quality is related to their expectations or requirements and one of the key determinants of consumers' perception is meeting their requirements. An experimental study conducted by Kim et al. [70] investigated the relationship between consumer expectation and perception in e-commerce. The empirical findings suggested that consumer expectation has a positive influence on consumer perception and the fulfillment of consumers' expectation is essential to improve consumers' perception of quality. The gap between what service providers delivered and what consumers expected is the 'perception gap' in Figure 2.1. To address this gap, some works in the literature ask service consumers to give ratings to indicate their perception on the quality of a service [113, 157, 153, 60].

While consumers' perception of quality may be based on their QoS requirements, the exact usefulness or value of a service to them may be influenced by various factors such as the cost of service. This is reflected in Figure 2.1 by the 'value gap'.

Thus, to help consumers to select services that can meet their requirements as closely as possible, it is essential that the gaps discussed above are considered in evaluating the alternatives. This could be achieved by collecting QoS related data during the cycle, and using this data suitably in QoS assessment. For example, by collecting the expectations on $S1$ from its previous users, and the quality delivered to them, the likely quality that Alice will receive from $S1$ can be assessed more meaningfully and accurately, thereby minimising the potential mismatch due to the execution gap.

2.3 Requirements for QoS Assessment Method

The various gaps in the quality cycle discussed in Section 2.2 are significant and should be considered when developing an effective QoS assessment method. An accurate and

reliable assessment of QoS can help reduce the 'execution gap', so that consumers can be guided to select a service provider that is more likely to meet their requirements. To achieve this, however, several important research challenges need to be addressed. In this section, we identify these challenges and formulate a set of requirements that must be considered in developing a sound QoS assessment method. These requirements form the basis of the work described and developed in this thesis.

Heterogeneity in Consumers' QoS Requirements

Individual differences will exist among consumers in terms of their expectations of the quality of service. That is, a particular quality requirement for a QoS attribute may vary from one consumer to another depending upon its importance to the individual [110, 89, 126]. For example, high `availability` and fast `response time` of a stock trading service will be important for a stock trader, since any delay may cause financial losses. However, this may not be the case for someone trying to book a flight ticket because small delays can be tolerated. Thus, a QoS assessment method should be capable of the following:

R1. Service providers should be assessed per consumer request (or requirement), because different requesters may have different quality requirements, and a service suitable for one user may not be suitable for another.

Multiple Levels of QoS

Service providers may adopt some policies to provide their services with different levels of quality to accommodate different consumers' QoS requirements [24, 33]. Also, service differentiation is generally used to deal with the complexity of resource management [84]. Shercliff et al. [127] pointed out that consumers may be classified into different QoS classes (or levels), each of which represents a set of consumers with

a particular set of requirements. Consumers with similar QoS requirements will be treated in a similar way by the provider. The policy adopted in classifying and managing consumers in QoS classes will differ from provider to provider, and providers will not always publish details of their policy [84]. This presents a challenge to QoS assessment tools and reputation brokers that attempt to identify the likely future behaviour of a provider. If providers provide different levels of quality to consumers based on their expectation, it is not justifiable for assessment approaches to consider all historic data for a particular provider as equally relevant when making assessment [27]. Such assessment would produce a result which would represent the average performance of the provider over consumers in all service classes, rather than the performance of the provider in terms of service consumers who are in the same service class as the one requesting the assessment. It is therefore essential that a QoS assessment method should be capable to handle the following:

R2a. The dynamicity of service behaviour over time.

R2b. Contextual information that leads to the provision of a certain level of quality.

R2c. Multiple levels of quality offered by a single service.

Reliability of QoS Verdicts

An important issue that needs to be considered when assessing QoS is the reliability of a verdict. Intuitively, the more historical data used in an assessment and the more consistent the data is, then the more confidence one would have in the verdict from that assessment. Unfortunately, it cannot always be assumed that we would have enough data to produce a reliable verdict, e.g. when a new service starts [123] or an expectation-based approach is adopted [27]. This is particularly important for QoS assessment involving multiple attributes: if the attributes are assumed to be asynchronously monitored [148, 66, 130], yet the QoS assessment method only considers the synchronous subset, one would be left with a very small fraction of the data to use, as shown in Figure 2.2.

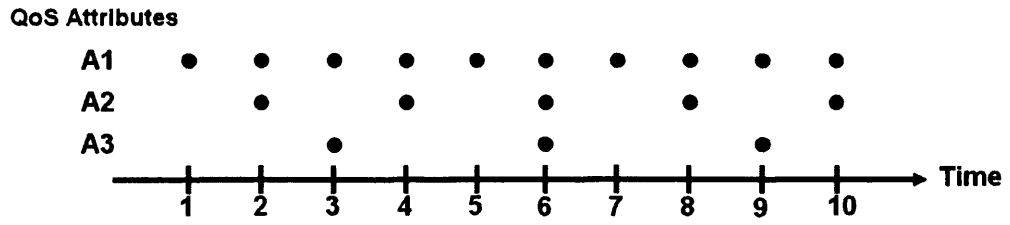


Figure 2.2: Asynchronous data over ten service provision instances

Here, we assume that A_1, A_2 and A_3 are collected every 1, 2 and 3 units of time, respectively. As can be seen, only one instance out of ten (at time 6) is usable for a QoS assessment that considers asynchronous data only when assessing multiple attributes. So, a good QoS assessment method should be capable of the following:

R3. Providing the confidence of a QoS assessment for every QoS verdict.

Multiple QoS Attributes

To deal adequately with services that offer multiple packages of quality, QoS assessment should be conducted based on collective rather than separate attributes. This is so that the possible grouping of quality levels that exists across multiple attributes can be captured and identified. For example, for a particular service, a high level of `availability` may be linked with a low level of `response time` and vice versa, due to some of the service provision policies adopted by the service provider [33]. If we have a consumer asking for high `availability` and high `response time`, the methods that assess multiple attributes individually may wrongly suggest that the service is capable of delivering the required service level. Thus, in more complicated service provision scenarios, where some grouping of quality levels exists across multiple attributes, calculating QoS for each attribute individually can result in an incorrect

assessment. To handle such scenarios adequately, a QoS assessment should meet the following requirement:

R4. QoS assessment should deal with multiple attributes collectively.

2.4 Survey of Existing QoS Assessment Methods

A plethora of computational QoS assessment methods has appeared in the last few years, each with its own characteristics and using different technical solutions. In this section, we analyse these methods using the set of requirements identified in Section 2.3 as our criteria, and we select methods that meet at least one of those requirements. We then classify current QoS assessment methods in terms of the data they use. Broadly, there are three types of data that may be used to assess QoS: service providers' advertisements, monitored QoS data, and user ratings. Different issues need to be considered in QoS assessment and these issues vary from one data type to another. In the following sections, we survey methods for QoS assessment classified by the data they use.

2.4.1 Service Provider Advertisements

QoS assessment methods that use service providers' advertisements assume that service providers publish their QoS data alongside the functional description of their services in a service registry such as Universal Description Discovery and Integration (UDDI) [100]. The methods using this approach attempt to assess the degree to which the advertisements of service providers match consumers' QoS requirements.

Certification Approach

This approach requires a certified reference about the quality of a service from third-party. Such information is obtained and made available in a central service repository

such as UDDI, and can be used by service providers to certify their capability and by service consumers to verify a provider's claims. The proposed methods in [110] and [124] adopt this approach.

Ran [110] proposed a QoS method which takes into account the consumer's QoS requirements in quality assessment. In this method, the QoS offered by the service provider is included with the functional aspects of the services that are being advertised in the service registry. To establish the consumer's confidence in the provider advertisements, a certification approach is introduced to certify QoS claims by providers and verify these QoS claims for consumers. In this method, the assessment results in the production of a certificate, a copy of which is stored with the third party. Such certificates may be referred to by the provider when advertising their services.

The authors in [124] detail and implement the certification process which has been introduced by Ran in [110]. They conducted a study in which a QoS broker (a certificate issuer) generates and executes a set of test cases which aim to test the quality of a service. Two verification techniques are used in this method. First, syntactic and semantic verification of service interface description is conducted. Second, a monitoring tool is used to compute the QoS values and compare them with what the service provider has claimed to deliver. This method, however, did not detail how the consumer's requirements and providers' offers are matched.

The QoS assessment in [110] and [124] resulted in the production of a certificate that can be used by consumers to select their preferred services. While the certificate may help establish consumers' confidence in advertised QoS, the level of performance delivered by any provider is likely to change over time [60]. Further, it is unlikely that the certifier will be using the same network configuration as the potential consumer of services [17], so the level of service verified by the certification process may be a poor indication of the actual level of service that will be received by an individual consumer.

Ontological Approach

This approach considers the importance of semantic information in QoS assessment. It relies on some predefined ontologies to provide terminology and formal semantic description of various aspects of QoS attributes. For example, each QoS attribute may be described in the ontology in terms of its type, measurement unit, and its correlation to other attributes.

Tian et al. [136] proposed a Web Service QoS (WS-QoS) framework which allows consumers to specify their QoS requirements and providers to publish different classes of their services (i.e. multiple levels of quality for the same service). They used the WS-QoS as a broker between consumers and providers to determine which offer best matches consumers' requirements. To keep QoS offers from providers up-to-date, the WS-QoS broker enquires of potential providers to get current QoS offers only at the time of an assessment request, this ensures availability of offers.

The QoS-IC framework proposed by Taher et al. [134] considers QoS assessment over multiple attributes. That is, their framework takes into account the semantic relationship between different QoS attributes in assessment. The authors introduced a QoS constraint model to establish and define association among QoS attributes as a set of constraints. An example constraint is that if a service is scalable, then it will be available [86] (e.g. $\texttt{scalability} \geq 0.5 \Rightarrow \texttt{availability} = 1$). When a consumer issues a service request which contains his QoS requirements, a set of services that satisfy the required functionality are retrieved and each offer is checked for its consistency with some predefined constraints before making the assessment. The matchmaking is then applied between QoS requirements and the consistent QoS offers w.r.t. the predefined constraints to choose the offer that best matches the consumer requirements.

The authors in [143] extended the Web Services Modelling Ontology (WSMO) [115] to annotate service descriptions with QoS data. The WSMO-QoS ontology is then used in their method for QoS assessment. In their approach, if a consumer does not state his re-

quirements on a particular QoS attribute, then the default preference will be used, based on a value that would normally be preferred by a consumer. For example, low price and high reliability are normally preferred, hence are assumed to be the consumer's preferences. If a requirement is given, then the consumer would require the delivered quality to be as close to his requirements as possible. In this method, a matrix is constructed as follows:

$$M = \begin{pmatrix} r_1 & r_2 & r_3 & \cdots & r_m \\ q_{11} & q_{12} & r_{13} & \cdots & q_{1m} \\ q_{21} & q_{22} & r_{23} & \cdots & q_{2m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ q_{n1} & q_{n2} & q_{n3} & \cdots & q_{nm} \end{pmatrix}$$

where r_j represents the consumer's requirements for attribute A_j and q_{ij} represents the quality offered by service S_i for attribute A_j. For uniformity, values in this matrix are normalised. In addition to QoS requirements, the consumer is asked to provide a weighted value for each attribute to indicate how important the attribute is. The weighted values are applied to the matrix and the final verdict for each service is calculated by summing up the values of each row.

The authors in [87] proposed a Web Service Procurement (WSP) framework which uses mathematical constraints to specify QoS offers and requirements. In this model, specifications of QoS offers and requirements are expressed as a bounded range of values. To assess how well QoS offers match the consumer's requirements, QoS offers and demands are treated as spaces and constraint programming is used to check whether the demand space is included in the offer space. For example, if a service offers $availability$=90, then it is conformant to the requirement $availability \geq 85$. The QoS offers and demands are expanded to allow more expressive time-dependent specifications in the later work [88]. That is, QoS offers and demands are subject to validity periods. This adds more complexity to the role of QoS assessment, since checking the conformity between QoS offers and demands necessitates considering their validity periods.

The ontology-based methods assess QoS based on a degree of matching between consumer requirements and service offers. In so doing, a complete trust in QoS offers is assumed. In practice, however, service providers may overestimate their capability in advertisements, either intentionally or unintentionally [60]. For example, a provider may exaggerate his QoS offers to attract consumers. So, the credibility of service providers for their advertisements must be taken into consideration in QoS assessment.

Trust-aware Approach

In this approach, a QoS advertisement is assigned a trust score to indicate how reliable a service provider is in providing the promised quality level. In common with the certification approach that has been discussed earlier, both approaches rely on an external third party to provide a trust score to certify the reliability of an advertisement. But the two differ in that the certification approach derived such a score statically before service registration, whereas the trust-aware approach does so dynamically during service consumption by monitoring the provider's performance over time via user feedback [151] or by use of dedicated monitoring tools [142].

Maximilien and Singh [89] developed a multiagent framework (Web Services Agent Framework - WSAF) based on an ontology for QoS and a trust model. The ontology provides a basis for service providers to advertise their offerings and consumers to express their requirements, and for service deliveries (monitored as user ratings) to be gathered and shared. These monitored measurements are used to provide an empirical basis on which to build consumers' trust (or confidence) in the advertised values. In an extension to the WSAF framework [91], the issue of statistical and ontological attribute dependence is addressed. In a similar work, a reputation manager was introduced by Xu et al. [151] to assign reputation scores to the services based on consumers' feedback. In this work, it is assumed that consumers will provide an honest appraisal of the service they have received. However, in some instances, consumers may deliberately mislead the reputation manager by providing inaccurate information [60]. The

problem of dishonest feedback is widely acknowledged in trust and reputation systems [25, 145, 154, 160, 85]. In [142], for each service, based on collected quality data relating to trustworthiness and credibility, a time series forecasting technique is used to predict its future quality conformance level, and a simple additive weighting method is used to calculate the final QoS value. This work was extended recently in [141] to improve the accuracy of quality estimation by developing a framework in which the reliability of rating is considered and unfair ratings are detected and filtered out.

All the methods discussed so far recognise and consider consumers' QoS requirements so as to identify a level of service that a consumer wants to receive. These methods attempt to calculate a degree of matching between the required and offered QoS. Additionally, to have more confidence on service advertisements, these methods produce trust values derived from user ratings to indicate how reliable a service will be in delivering what has been advertised. However, in a dynamic service provision environment, discrepancies between advertised and delivered QoS can arise due to several reasons. First, the service provider may implement some policies to deliver multiple QoS levels to different consumers based on their requirements and the policies may not be published. Second, the QoS is dynamic in nature and may be affected by various factors. For example, excessive requests may lead to the provider's resources becoming overloaded and consequently decrease the quality delivered to the consumer. So, the bigger the difference between the advertised and the actual delivered QoS to the consumers is, the less meaningful the verdicts from these methods will be. A set of methods in the literature attempt to address this issue by assessing services based on their historical QoS data rather than QoS advertisements. In the next section, we discuss these methods in detail.

2.4.2 Monitored QoS Data

In an environment where it is assumed that a large centralised information broker is practical and whose trust may be assured, the use of objective monitored data is an

effective quality assessment approach. The basic premise underlying this approach is to collect the QoS information from monitoring tools for different attributes, which can reside either locally on the server providing the service [5, 152], locally on the consumer side using the service [22], or at some point in between [83, 67], and then to aggregate them in some way to give an overall verdict for each attribute. In this approach, the aggregated value is used as a prediction of the delivered QoS that might be received by a consumer with respect to the attribute being assessed.

Generally speaking, there are two approaches to QoS assessment using monitored data, non-selective and selective. The non-selective approach simply uses all the historical QoS data in assessment, while the selective approach uses part of the historical QoS data determined by various mechanisms. In the following sections we discuss these two approaches in detail.

Non-selective Approach

This approach implies that the assessment method considers all historical data as relevant. The idea of assessing the quality of a service using historical QoS data has been described in [81, 22]. In the UX architecture [22], a QoS broker, an extension of UDDI, is deployed between consumers and providers of web services. Monitored QoS data regarding `response time`, `cost`, `network bandwidth` and `availability` of each service are collected and stored in a centralised database. The basic concept is to estimate the QoS that a consumer may receive based on the deliveries to other consumers by aggregating all the collected QoS information. In a similar method, Liu et al. [81] proposed an extensible and dynamic QoS assessment method about how to combine different QoS metrics to get a fair overall ranking for a service. Monitored QoS data, and other technical and business attributes such as execution price, execution duration, and transaction are collected and used in the assessment.

The methods for assessing QoS described above suffer from two fundamental weak-

nesses. First, they do not require the consumer to state any QoS requirement. It is assumed that a consumer would always want to maximise or minimise QoS for a particular attribute. This assumption may be justified for some attributes. For example, a consumer may wish to have a service that has the highest `availability` and the lowest `latency` possible, as higher `availability` and lower `latency` will always lead to a better service. However, for other attributes a consumer may prefer a value which is not at the extremes of the scale. Such preferences may be due to some specific constraints. For example, limited by physical capability, a consumer who is using a mobile device to access a streaming video service may not wish to have a `frame rate` above a certain threshold, as this could overload the mobile device making the video no longer watchable [9].

So, perceptions of QoS by different consumers may be affected by their requirements or expectations [70, 103, 158]. That is, a certain delivered QoS may be good enough for one consumer but not for another. Second, these methods do not consider the contextual information that leads to the provision of a certain level of quality. For example, to deal with the complexity of resource management, service providers may adopt policies to provide different levels of quality to their consumers based on their requirements [127]. Ignoring such contextual information when assessing QoS may lead to incorrect QoS assessment [27]. So, except for the case where service providers deliver a single level of quality to all consumers, regardless of their requirements, the non-selective approach can produce incorrect assessment. In the next section, we discuss more advanced techniques to address this issue.

Selective Approach

In contrast to the non-selective approach, the methods of the selective approach assume that not all collected data may be relevant to a QoS assessment request. More advanced techniques use various heuristics to determine which data is relevant to a QoS assessment. Broadly, the methods of this approach can be classified into two groups:

collaborative filtering-based methods and expectation-based methods. While the former pays more attention to the contextual factors (e.g. network barrier, consumers' locations, etc.) that may affect the level of QoS perceived by consumers, the latter specifically focus on the variation in service provider behaviour in terms of delivered QoS data which may stem from the difference in the consumer's QoS requirements. In the next sections, we discuss these methods.

Collaborative Filtering-based Methods

Inspired by recommender systems in e-commerce, these methods emphasise on the importance of taking contextual information into account in QoS assessment. Conceptually, a recommender system is similar to QoS assessment in that it provides consumers with information to help them decide which product to purchase or which service to use [45, 47]. The most widely used technique for recommendation is based on collaborative filtering, which can be defined as the process in which users collaborate to assist each other to perform filtering by recording their experiences [112]. Two types of algorithm exist for collaborative filtering, memory-based [122, 77] and model-based [75]. The former aims to identify a set of users that share similar interests (e.g. users who have purchased the same items from Amazon). A set of items is then recommended to a user based on aggregated ratings of items from these similar users. A memory-based algorithm operates over the entire user database to establish similarity and analyse user behaviours to make recommendation. A model-based algorithm, on the other hand, attempts to establish a model that consists of a number of classes and similar users are grouped together in a class based on their rating behaviour. These pre-computed models are used to make recommendations by simply linking a given user to one of the classes.

Recent work [125, 162, 116, 21] has been undertaken to apply collaborative filtering to QoS assessment. In collaborative filtering-based methods, QoS is predicted for an assessment requester based on the historical QoS data provided by consumers who

have similar experience of other services. The basic assumption of these methods is that those consumers, who have received similar quality for some services, will receive similar quality for other services. Shao et al. [125], employed a collaborative filtering mechanism to mine service consumer similarity and to predict QoS based on collected QoS data. In their method, the similarity between two consumers' historical experiences is calculated using the Pearson Correlation Coefficient (PCC). Then, the QoS of a target service is predicted based on the similarity. Since the PCC method may overestimate the similarities of service consumers, especially with sparse data (i.e. over a small amount of consumers), WSRec [162] combined user-based and item-based collaborative filtering using a similarity weight. In the user-based collaborative filtering, PCC is employed to define the similarity between two consumers based on the services they commonly used. In the item-based collaborative filtering, on the other hand, PCC is employed to define the similarity between the services instead of consumers. The WSRec provides a confidence score to indicate how accurate the predicted QoS value is. The authors of [21] argued that the methods given in [125] and [162] do not consider the difference between the characteristics of objective QoS data and subjective user ratings, and therefore their prediction is inaccurate. To redress this problem, they proposed RegionKNN, a hybrid model-based method and memory-based method. A region model is introduced to compress QoS data by clustering users into different regions based on physical locations and historical QoS similarities. Based on the region model, a refined nearest-neighbour algorithm was proposed to predict QoS values.

The methods of this approach can be viewed as attempting to address the gap between delivered and perceived QoS in the quality cycle. However, these methods suffer from two disadvantages. First, their assessment is not based on the current consumer's requirements. Second, these methods may give misleading assessment when a provider delivers multiple levels of quality to consumers based on their requirements. This is because it is possible for two consumers in the same context (in the same region, for example), but having different requirements, to receive different QoS levels. Contextual information such as consumers' requirements or expectations may help explain a

delivered quality value and produce more confidence in assessment methods [46]. This issue is considered and addressed by the methods described in the next section.

Expectation-based Methods

The basic assumption of this approach is that the variation in service providers' behaviour in terms of delivered QoS data stems from the difference in consumers' QoS requirements. Thus, to produce a more accurate assessment, the requirements of the assessment requester must be captured and considered alongside the actual QoS data in future assessment. As such, these methods can make a more personalised QoS assessment. The basic idea of these methods is that instead of using the entire historical data in QoS assessment, a portion of the history is selected with respect to the assessment requester's requirements or expectation.

The expectation-based approach was introduced by Deora et al. in [27] to collect user expectations as well as ratings from the users of a service. The QoS expectation is obtained and then used to determine a subset of historical data that is similar to the requester's expectation. Following that, a standard aggregation is carried out. Sherchan et al. [126] concur with [27] on the importance of using expectation in QoS assessment, but they use objective monitored data rather than subjective ratings. The work in [127] addresses the problem of QoS assessment when service providers offer different levels of quality for different consumers. However, their Multiple Quality-Space Mapping (MQSM) method is limited to considering possible levels within a single attribute only. The method in [149] takes multiple QoS attributes into account and predicts service capability in various combinations of consumer requirements. The authors use a Bayesian network model to predict whether a service can deliver the expected QoS. This model uses monitored data to compute the compliance (i.e. conformance) between the QoS delivered by a service provider and the QoS required by a service consumer. The service capability is then inferred based on the compliance value and the Bayesian network is updated using the outcome of influence. However, their method is unlikely

to perform well in practice, because it implicitly assumes that QoS data is collected synchronously across multiple attributes. In practice, however, QoS data is more likely to be collected asynchronously, as we previously argued. All the methods discussed so far, therefore, do not handle multiple attributes adequately, especially when these attributes are monitored asynchronously.

The methods of this approach can be viewed as attempting to address a number of gaps in the quality cycle. By considering consumer's requirements or expectations in assessment, they deal with the alignment gap. The gap between advertised and delivered QoS is filled by using monitored QoS data. Finally, adhering to conformance view of quality means that these methods address the perception gap between delivered and perceived QoS. Other methods in the literature address the perception gap differently by asking service consumers to provide their feedback as ratings at the end of each service provisioning. The ratings implicitly represent the level of QoS the consumers 'believe' they have experienced. We discuss the methods using this approach in the next section.

2.4.3 User Ratings

Reputation and trust are similar to quality of service in that they are also an assessment of someone's performance using historical data, so are broadly relevant to QoS assessment. Typically, reputation and trust methods seek to establish the quality of a service by gathering ratings from consumers who have used the service. In this approach, the consumer's stated rating implies the quality they believe they have experienced. Trust is defined as the extent to which the consumer is willing to depend on the provider in a given situation [35]. Reputation, on the other hand, can be considered as a collective measure of trustworthiness based on the history of interactions [79]. While trust is subjective, reputation is more objective because it is usually a collective opinion of the whole community.

A large number of methods in the literature define QoS as the reputation of a service

provider [113, 157, 153, 65, 150]. Typically in these methods, the observations of service providers' behaviour are obtained from service consumers over time as ratings. Usually, binary ratings are used in probabilistic approaches such as bayesian analysis [62] or beta distribution [59] to produce a QoS verdict, while aggregation mechanisms such as simple [27] or weighted average [96, 153] are used for numerical ratings.

Probabilistic Approach

In this approach, service consumers are asked to provide their experience as a boolean value to reflect how good service providers are in delivering their promises. The consumer feedback in this approach has two values: 1 for a successful interaction and 0 for a unsuccessful one. This approach attempts to determine how likely it is that a service provider will deliver the expected performance in future interaction. This probability is used as a trust value for the provider.

A notable class of trust models based on binary rating systems are those that calculate trust values using probability density functions (PDFs) [59]. $eBay$ auctions [1] are one such example. Online reputation mechanisms used by $eBay$ are implemented as a centralised rating system so that $eBay$ users can report about the behaviour of one another in past transactions via a rating and leaving textual comments. In so doing, users can learn about the past behaviour of a given user to decide whether he is trustworthy to do business with. An $eBay$ user can rate its partner after an interaction on the scale of -1, 0, or +1, respectively a negative, neutral and positive rating. The ratings are then stored centrally and the reputation value is computed as the sum of those ratings over six months. Thus, reputation in this model is a global single value representing a user's overall trustworthiness.

SPORAS [157] is an evolved version of this kind of reputation model. In this model, each consumer rates its partner after an interaction and reports its ratings to a centralised database. The received ratings are then used to update the global reputation values of the rated consumers. However, instead of storing all the ratings, each time a rating is

received it updates the reputation of the involved party using an algorithm that satisfies some policies. For example, users with very high reputation values experience much smaller rating changes after each update. In addition, SPORAS also introduces a reliability measure based on the deviation from the mean rating value. For instance, a high deviation value can mean that rating providers have very different opinions about the quality of a service. When rating providers deviate a lot from the mean rating value for the service, the SPORAS system assigns a very low reliability to the provider's rating and a high reliability if the provider rating is close to the mean rating value. This reliability value is used in the service reputation calculation. For each service, a reputation and a reliability value are made available to other consumers globally. Also, in SPORAS, the reputation value of a service and its reliability are discounted over time as a new rating is received. Therefore, SPORAS can adapt to changes in service behaviour according to the latest rating. Histos [157], an extension of SPORAS, is a more personalised reputation system where reputation depends on who makes the quality assessment request. Although all these models are effective in their context, they only consider the trustworthiness of a service provider in one dimension (i.e. considering a single attribute). This is not suitable for a service (or product) that has multiple QoS attributes, for example, when a consumer considers a web hosting service to have a good availability, but a poor access delay.

TRAVOS [135] is a trust model that is built upon probability theory and is based on observations of past interaction between service providers and consumers. In this model, the outcome of an interaction is simplified into a binary rating (i.e. 1 for a successful interaction, 0 for an unsuccessful one). Using binary ratings allows TRAVOS to make use of the beta family of probability density functions (PDF) to model the probability of having a successful interaction with a particular given service provider. This probability is then used as that provider's trust value. In addition, using PDFs, TRAVOS also calculates the confidence of its trust values given an acceptable level of error. If the confidence level of a trust value for a provider based on the current consumer observations is below a predetermined minimum level, TRAVOS will use other consumers observa-

tions about the provider's past performance, which are shared in the form of frequencies of successful and unsuccessful interactions (e.g. (10,15)). This then allows the current consumer to calculate the probability that the observations of other consumers support the true behaviour of the provider that he or she has observed within a reasonable margin of error. This probability will be used by the current consumer to weight the impact of other consumer observations on future decisions. However, TRAVOS's simplified representation of interaction ratings (0 or 1) is rather limited and is not suitable for a wide range of applications, for example, if we wish to classify providers into not good or bad, but good, bad or average.

Jøsang et al [59] proposed a binomial Bayesian reputation system which allows ratings to be expressed with two values, as either positive (e.g. good) or negative (e.g. bad). The disadvantage of a binomial model is that it excludes the possibility of providing ratings with graded levels (bad - mediocre - average - good - excellent). Principally, binomial models are unable to distinguish between polarized ratings (i.e. many very bad and many very good ratings) and average ratings [61]. In [58, 62] Jøsang et al. presented a type of reputation system based on the Dirichlet probability distribution which is a multinomial Bayesian probability distribution. The representation of reputation systems based on the Dirichlet distribution allows graded ratings to be directly expressed and reflected in the derived reputation scores. In other words, the model supports multinomial user ratings rather than only binary ratings as in [135]. This system computes the expected reputation scores by combining previous interaction records with new ratings. Reece et al. proposed a probabilistic model of trust that deals with multiple correlated dimensions [111], and they used Dirichlet distribution to estimate trust from the direct experience of an agent as well.

Aggregation Approach

In contrast to the probabilistic approach, this approach asks service consumers to provide their feedback on a numerical scale (e.g. from 1 to 5). The collected feedback

from consumers is then aggregated using some functions, such as arithmetic mean, to predict the expected performance of a provider in a future interaction. Examples of this approach are the methods proposed in [117, 55].

Regret [117] is a decentralised reputation model in which each consumer is able to evaluate the reputation of service providers by himself. To do so, each consumer rates a partner after every interaction and records his ratings in a local database. The relevant ratings will be queried from this database when trust evaluation is needed. The trust value derived from those ratings is termed *direct trust* and is calculated as the weighted means of all ratings. Each rating is weighted according to its time stamp. That is, a more recent rating is deemed to be more relevant and is weighted more than those that are less recent. In doing so, Regret was able to adapt to any change in a service provider's behaviour. This strategy is effective in encouraging service providers to be consistent over time in delivering their services. Moreover, it will help to detect and discount a malicious provider that builds a reputation by performing honestly initially, and then starts "milking" the attained reputation by cheating on a number of transactions. However, if recent behavior is assigned a very high weight, then the provider that has high reputation will lose the attained reputation after a few misbehaviours and vice versa. Additionally, in Regret, consumers are assumed to be willing to share their opinions about service providers. Based on this, a *witness reputation* component is developed alongside a method for aggregating witness reports, taking into account the possibility of dishonest reports. The operation of this component depends on the social network built up among the consumers. The social network is used by the Regret system to determine the relationship between individual consumers. In particular, Regret uses the social network to find witnesses, to decide which witnesses will be consulted, and how to weight those witnesses' opinions. Like SPORAS, Regret also provides a reliability value for each trust value to represent its predictive validity. The reliability value is calculated from two reliability measures: the number of ratings taken into account in producing the trust values and the standard deviation of these ratings. Zhang and Cohen [160] used the Chernoff Bound [96] to determine the minimum number of

ratings needed in order to be confident about a trust value.

The authors in [55] argued that Regret does not show how each consumer can build the social network on which Regret heavily depends to find witnesses and thus, the *witness reputation* component is of limited use. They reused the interaction trust component of Regret, but their Fire model overcomes the limitation of Regret by employing a referral process in which consumers help each other to find witnesses based on their expertise. In Fire, a variant of the referral system proposed in [155] is used to find such witnesses. To do so, consumers cooperate with each other by giving, pursuing and evaluating referrals (a recommendation to contact another consumer). Each consumer in the Fire model maintains a list of acquaintances (other consumers that he knows). Thus, when looking for a certain piece of information, a service consumer can send a query to a number of acquaintances who will try to answer the query if possible or, if they cannot, they will send back referrals pointing to other consumers that they believe are likely to have the desired information. Similar to Regret, the reliability value is based on the rating reliability and deviation reliability to counteract the uncertainty due to instability of services.

The ratings-based methods (probabilistic and aggregation) can be viewed as attempting to address the perception gap between delivered and perceived QoS in the quality cycle. However, while ratings give the consumer's perception of service quality, they do not help to indicate the actual level of service delivered by the provider. Also, these methods do not consider consumer's expectation as part of the context, and thus do not help identify the reasons behind the ratings given by the consumers [27]. Additionally, ratings are plagued with the issues of subjectivity, collusion, identity and maliciousness [60]. It has been observed that due to fear of retaliation most ratings tend to be biased and unrelated to the actual delivered level of quality [60]. Various methods in the literature are proposed to handle these issues. For example, the methods reported in [25, 145, 154, 160, 85] employ different techniques to detect and remove unfair ratings.

Table 2.2: The meaning of symbols used in Table 2.3

Dimension	Symbol	Meaning
Assumptions	SA	Single Attribute
	MA	Multiple Attributes
	T	Trustworthy
	N	Normalised
	D	Dependent
	A	Asynchronous
General	Empty	The requirement is not satisfied
	√	The requirement is satisfied
	N/A	The requirement is not applicable

2.4.4 Discussion

The comparison of QoS assessment methods reviewed in this section is summarised in Table 2.3 and the meaning of symbols used in the table is given in Table 2.2. The methods are analysed and compared in terms of the type of QoS data used in assessment, assumptions about QoS data, and how they satisfy the requirements listed in Section 2.3. We have three groups of methods based on the QoS data type: advertisement-based methods, monitoring-based methods, and ratings-based methods. The assumptions about the data used in assessment determine, to a large extent, the power and applicability of a specific assessment method.

The QoS data type and assumptions have a direct impact on the capability of a method to satisfy requirements **R1-R4**. For example, the advertisement-based methods fail to satisfy requirement **R2a** because these methods require a service provider to publish QoS data in service registration and this may not be updated regularly. The dynamic behaviour of a service provider is captured explicitly by monitoring-based methods and implicitly by ratings-based methods. However, how **R2b** and **R2c** are satisfied by these methods depends on how they use the collected data, ratings or monitored data. For example, monitoring-based methods employ collaborative filtering and expectation-based

Table 2.3: Comparison of the reviewed QoS Assessment Methods

Data Type	Approach	Reference	Assumptions				QoS Assessment Requirements					
			SA		MA		R1	R2a	R2b	R2c	R3	R4
			T	N	D	A						
Service Provider Advertisements	Certified	Ran'03 [110]				N/A	√				√[a]	√
		Serhani'05 [124]				N/A	√				√[a]	√
	Ontological	Martin'03 [87]	√			N/A	√					√
		Tian'04 [136]	√			N/A	√			√[b]		√
		Taher'05 [134]	√		√[c]	N/A	√					√
		Wang'06 [143]	√			N/A	√					√
	Trustworthy	Max'04 [89]			√[c]	N/A	√	√[d]			√[e]	√
		Xu'07 [151]				N/A	√	√[d]			√[e]	√
		Vu'09 [141]			√[c]	N/A	√	√[d]			√[e,f]	√
Monitored QoS Data	NonSelec.	Zhou'03 [22]	√					√				√[g]
		Liu'04 [81]	√					√				√[g]
	CF-based	Shao'07 [125]	√					√	√[h]			√[g]
		Zheng'09 [162]	√		N/A	N/A		√	√[h]			
		Chen'10 [21]	√		N/A	N/A		√	√[h]			
	Exp-based	Sherchan'05 [126]	√				√	√	√[i]			√[g]
		Shercliff'06 [127]	√	√	N/A	N/A	√	√	√[i]	√		
		Wu'07 [149]	√				√	√	√[i]	√		√
User Ratings	Probabilistic	eBay [1]	√		N/A	N/A		√[j]	√[k]			
		Zacharia'00 [157]	√		N/A	N/A	√[l]	√[j]	√[m]		√[n]	
		Zhang'06 [160]			N/A	N/A		√[j]	√[o]		√[f]	
		Teacy'06 [135]				N/A		√[j]			√[f]	√
		Reece'07 [111]	√		√[c]	N/A		√[j]				√
		Josang'09 [62]	√		N/A	N/A		√[j]		√[p]		
	Aggregated	Sabater'01 [117]				N/A		√[j]	√[q]		√[r]	√
		Huynh'06 [55]				N/A		√[j]			√[r]	√

[a] The method maintains uncertainty by enforcing the service provider to provide a certificate.

[b] The method allows service providers to advertise multiple levels of QoS.

[c] The method considers semantic correlation between QoS attributes.

[d] The dynamicity of service provider behaviour is considered in reliability measures.

[e] The method uses reputation and trust to express the reliability of provider advertisements.

[f] The method considers the number of user feedback in reliability computation.

[g] The method assesses QoS based on individual attributes.

[h] The contextual information is considered implicitly by applying collaborative filtering techniques.

[i] The contextual information is considered explicitly by using a portion of history selected with respect to the current user request.

[j] The dynamicity of service provider behaviour is implied in consumer ratings.

[k] The method allows users to provide short text description with their ratings.

[l] The users do not state their requirements explicitly, instead they are extracted implicitly from recent transactions.

[m] The recent ratings are given more weight in trust/reputation computation.

[n] The method takes into account the reputation of rater in trust/reputation computation.

[o] The rating's timestamp is considered in detecting unfair ratings.

[p] The method supports non-binary ratings to allow consumers to rate a service in different levels (e.g. bad - average - good).

[q] The method uses social relations between agents in trust/reputation computation.

[r] The method uses ratings' size and variation for confidence calculation.

methods to deal with contextual information that can help explain a certain level of quality (**R2b**). These two techniques are used to determine and select part of the historical data that is relevant to a particular assessment request. It is obvious from Table 2.3 that the methods that assume trust in QoS data pay no attention to confidence measures (**R3**). Some assumptions, in fact, are applicable only to a certain type of data. For example, an asynchronous assumption is only meaningful when dealing with multiple QoS attributes whose quality data is collected from monitoring tools, not provided as ratings or advertisements.

The advertisement-based methods in Table 2.3 handle uncertainty with the advertised values by either enforcing that a service provider provides a certificate, or relying on the historical data to build up a reputation verdict, to demonstrate the reliability of an advertisement. All these methods take a personalised approach in assessment (**R1**) by considering consumers' QoS requirements. In their computation, these requirements are taken as a reference to assess the best service for consumers. Although the advertisement-based methods consider multiple attributes in their assessment (**R4**), only the methods proposed in [134, 89, 141] consider the relationship between these attributes. In general, the advertisement-based methods are not suitable in an open and dynamic environment. This is because the behaviour of service providers may change over time, yet their advertisement may not be updated regularly.

The monitoring-based methods, on the other hand, are able to capture the dynamic behaviour of a service provider over time (**R2a**). This is because these methods assume that the QoS data is collected during service provision. However, these methods do not deal adequately with multiple attributes (**R4**), especially when the qualities of these attributes are monitored asynchronously. Our analysis suggests that these methods assess multiple attributes either by assuming that they are independent, or by assuming that their data is synchronously collected. The non-selective methods given in Table 2.3 do not consider consumers' requirements in assessment (**R1**), assuming that either all consumers prefer to receive the highest possible level of quality, or a service provider

delivers a single level of quality to all consumers regardless of their requirements. Only two methods consider multiple levels of quality for a single service (**R2c**), but they do not give an indication of the reliability (confidence) of their assessment (**R3**).

The ratings-based methods in Table 2.3 emphasise the importance of confidence measure in producing trust/reputation values (**R3**). The possible reason for that is these methods need to address the issues of subjectivity, collusion and maliciousness of user ratings. Almost all these methods use two indicators to compute confidence: the number of ratings and deviation among the ratings. These methods, however, do not take into account consumers' requirements (**R1**) and ignore the rationale behind ratings by using the opinions of the members of the whole community equally in assessment.

In general, it is clear from Table 2.3 that none of the reviewed QoS assessment methods fully meet the requirements listed in Section 2.3. Our aim in this thesis is to develop a method that satisfies all the listed QoS assessment requirements.

2.5 Summary

In this chapter, we have addressed three key points. First, we considered the quality paradigms adopted by different QoS assessment methods in the literature. Quality as reputation links quality to the consumer's perception of a service, while QoS as conformance links quality to the degree to which a service provider meets a particular QoS requirement. To effectively support a consumer in finding services that best meet their QoS requirements, we adopt the conformance view of quality in our work. That is, we assume that as the difference between the QoS delivered by the service provider and required by the service consumer decreases, the consumer perceived quality of that service increases.

Second, we described and analysed different aspects of QoS assessment in an open service provision environment. Our investigation concluded that, in order to produce an effective QoS assessment, several gaps in the quality cycle must be filled. That is,

the gaps between the level of service required by the consumer, offered by the provider, delivered by the provider and perceived by the consumer need to be considered and filled by a QoS assessment. To do so, we provided a set of requirements that must be considered when developing a good QoS assessment method.

Finally, we reviewed and analysed the most relevant QoS assessment methods to our work. We evaluated to what extent these methods satisfied and fulfilled the identified requirements. Broadly, the reviewed methods can be classified into three groups: advertisement-based, monitoring-based and ratings-based methods. To capture the dynamic behaviour of service providers, we use monitored QoS data in our work. Our investigation and analysis of monitoring-based methods suggested that they do not deal adequately with multiple attributes, especially when the qualities of these attributes are monitored asynchronously. Moreover, these methods make unrealistic assumptions: they either assess multiple attributes individually assuming they are independent, or collectively assuming their data are synchronously collected. In the following chapters, we develop a method that takes both accuracy and confidence into account when assessing QoS over multiple attributes.

Chapter 3

A Conceptual Model for QoS Assessment

The goal of this chapter is two-fold: to introduce a conceptual model for characterising the tasks associated with QoS assessment, and to use this model to guide the development of a specific method for QoS assessment described in this thesis. The conceptual model for QoS assessment is introduced in Section 3.1. Section 3.2 gives a definition of quality that is used in this thesis, followed by the formulation of our QoS assessment problem. Following the discussion and overview of QoS assessment methods in Chapter 2, we discuss specifically the shortcomings of existing approaches in dealing with multiple attributes in Section 3.3. Finally, we summarise the finding of this chapter in Section 3.4.

3.1 QoS Assessment Process

As we have discussed in the previous chapter, there are three types of data that may be used for QoS assessment: service advertisements, user ratings and monitored service provision data. In this thesis, we focus on how the quality of a service may be determined using monitored QoS data, as we are interested in automated QoS assessment. Note that such data is already being routinely collected in order to, for example, enforce service level agreement [67, 94, 95, 128] or optimise service provision perfor-

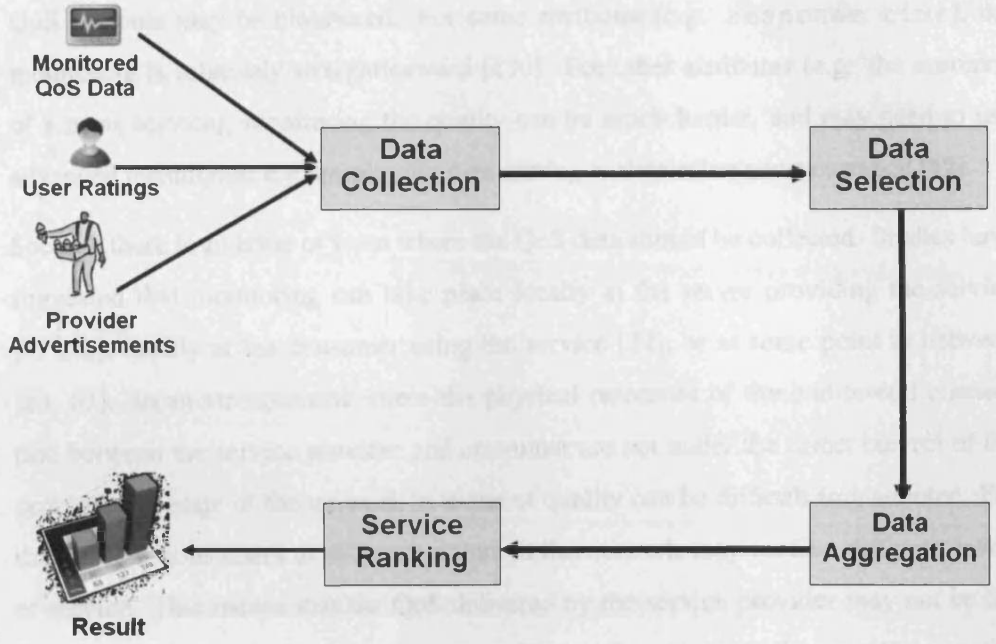


Figure 3.1: A Conceptual Model of QoS Assessment

mance [82, 6, 119, 74, 156]. Thus, it is reasonable and useful to consider how QoS assessment tools using the data collected by monitoring tools may be developed.

So, QoS assessment in our context may be broadly described as a function to determine, using the monitored historical service provision data, the likely quality that a consumer may get from a service provider. To discuss how effective QoS assessment methods may be developed, it is useful to understand what is involved in the process of QoS assessment in general first. As outlined in Figure 3.1, we consider a QoS assessment process involves four fundamental tasks: data collection, data selection, data aggregation and service ranking.

3.1.1 Data Collection

This is about how data relevant to the quality of a service may be obtained. There are a number of issues that need to be considered. First, we need to consider how a

QoS attribute may be monitored. For some attributes (e.g. `response time`), the monitoring is relatively straightforward [130]. For other attributes (e.g. the currency of a news service), monitoring the quality can be much harder, and may need to use advanced techniques, e.g. employing data mining to determine news currency [42].

Second, there is an issue of from where the QoS data should be collected. Studies have suggested that monitoring can take place locally at the server providing the service [5, 152], locally at the consumer using the service [22], or at some point in between [83, 67]. In an environment where the physical resources of the end-to-end connection between the service provider and consumer are not under the direct control of the provider, the state of the network in terms of quality can be difficult to guarantee. For this reason, consumers at different points in the network may receive different levels of service. This means that the QoS delivered by the service provider may not be the same as the QoS perceived by consumers. Thus, where to place the monitoring activities is important. While some attributes (e.g. `response time`) are best collected from the consumer side to capture the provider's performance as seen by the consumer, other attributes (e.g. `availability` and `reliability`) need to be measured and collected from the provider side to capture the provider's overall performance w.r.t. that attribute.

Third, granularity and units of measurements must be considered. For a given attribute, its quality can be measured and stored in different levels of granularity, for example, `access delay` may be measured in minutes, seconds, or milliseconds. For multiple attributes, different units of measurements may be used, for example, the `failure rate` and `availability` of services can be expressed as a percentage (e.g. 90%), whereas `access delay` could instead be an absolute value (e.g. 25 seconds). Moreover, the monitoring process may also capture some noise [48]. All these issues need to be considered and the collected data needs to be normalised in order for the data concerning different QoS attributes to be comparable and for the derived QoS verdicts to be meaningful.

Fourth, for multiple attributes, an additional issue needs to be considered. For example, multiple attributes may be monitored independently at different rates. An example of this is the `throughput` which may be collected per second (i.e. how many requests are served per second), whereas `availability` may be collected per hour (i.e. the percentage of up time per hour). It is worth observing that the data over multiple attributes may not be synchronously collected w.r.t. time, and may not be independent of each other either.

So, how QoS data may be monitored, collected and normalised can be complex in practice [130]. In this study, however, we do not consider such issues and simply assume that the data has already been collected and normalised. We assume that the QoS at any point on the link from the provider to the consumer is equal. However, we do not assume that QoS data for multiple attributes is synchronously collected as it is unrealistic and unlikely that this will be the case in practice [148, 66, 130]. We do not assume that QoS data in different attributes are independent of each other either, because it is quite possible that some quality patterns or groupings will exist, as our example in Section 1.1 has shown.

3.1.2 Data Selection

The data selection task is to determine which data should be selected for use in assessment. Not all collected QoS data may be relevant to a particular assessment request. Many issues need to be considered to determine which data should be selected for use in assessment. For example, if a service provider offers a service with several quality packages, then it is easy to see that combining the monitored data that are associated with different quality packages in assessing its quality could be misleading. Various techniques may be employed to determine which data should be selected for use in assessment. For example, Deora et al. [27] introduced expectation-based selection, where consumers are asked to state expected QoS levels as part of their assessment request, and only the data that has similar expectation to those stated in the request will be se-

lected and used. This approach was also adopted in [126, 127]. In our work, we use the expectation-based model [27]. That is, we record consumer expectations on quality alongside the actual quality monitored (see our QoS Model in Section 3.2), and then select and use only the QoS data that has similar expectations to those requested by the consumer in assessment.

3.1.3 Data Aggregation

Selected QoS data must then be aggregated to give an overall verdict on the quality of a service that the service provider is likely to deliver. How QoS data should be aggregated depends on the type of QoS data involved. For example, binary data (e.g. satisfied or unsatisfied) may be aggregated into a single verdict using beta probability density functions [59, 160], whereas numerical data may be aggregated based on simple or weighted average [27, 81, 125, 162], and forgetting or damping factors can be employed to help discount past performances [59, 157]. For multiple attributes, however, existing works largely assume that the data in each attribute may be aggregated individually, which, as we have explained in the Introduction (Section 1.1), can lead to an incorrect assessment. This is because assessing multiple attributes individually is not able to identify and capture the interaction among multiple attributes. For example, for a particular service, a high level of `availability` may be linked with a low level of `response time` and vice versa, due to some of the service provision policies adopted by the service provider [33]. If we have a consumer asking for high `availability` and high `response time`, the methods that assess multiple attributes individually may wrongly suggest that the service is capable of delivering the required service level. In our work, we propose to aggregate QoS data over multiple attributes collectively rather than individually.

3.1.4 Service Ranking

If the goal of QoS assessment is to help a consumer to choose a preferred one among those functionally identical but quality-wise varying services, then it is essential that we are able to rank a set of services somehow at the end of assessment. One obvious approach is to attempt to deliver a single numeric verdict for each service under assessment, and then rank the services under assessment based on their numerical order [81, 22]. Unfortunately, it is not always desirable or possible to derive a single verdict, for example, when the quality of each attribute must be considered and compared separately. Ideally and in practice, a service ranking function should consider a range of differentiating factors, for example, accuracy and confidence of QoS verdicts, prices of services and trust in service providers, and then rank services based on these factors. In such cases, more sophisticated solutions based on multiple criteria decision making principles [34] must be considered. For the purpose of this thesis, we rank services based on assessment accuracy and confidence only. We combine accuracy and confidence scores into a single numerical verdict for each service under assessment, and then rank the services based on this single numerical value.

3.2 A QoS Model

In this section, we give a definition of quality that we will use in this thesis, followed by the definition of our QoS assessment problem.

3.2.1 Definition of Quality

Different definitions of quality exist [27]. Broadly, as we have mentioned in Section2.1, quality can be linked to reputation, which refers to the perception of the service that a service provider builds over time about its intentions and norms [96], or linked to conformance, which indicates the degree to which a service provider meets (or conforms

to) a particular 'ideal' level. For example, an Internet service provider may be considered by a consumer as being able to offer good quality even though the user has not used the service, because the provider has the *reputation* for delivering a good service consistently over time. Equally, a consumer may consider the provider to offer good quality because it delivered the 4 MB internet access speed that the consumer has asked for, i.e., the provider *conforms* to what is required.

We adopt the conformance view of quality in our work, as we are interested in assessing the quality that a service provider can offer for a specific consumer. That is, we consider quality to be the difference between what is requested by a consumer and what is actually delivered to the consumer. More specifically, let $S(A_1, A_2, \ldots, A_m)$ be a service where each $A_i, i = 1, \ldots, m$, is a QoS attribute. Suppose that S is required to be delivered to a consumer with an expectation $\gamma = \{e(A_1) = \alpha_1, e(A_2) = \alpha_2, \ldots, e(A_m) = \alpha_m\}$, where $e(A_i) = \alpha_i$ represents the quality expected by the consumer on A_i. Suppose that during the service delivery $\{d(A_1) = \beta_1, d(A_2) = \beta_2, \ldots, d(A_m) = \beta_m\}$ is monitored, where each $d(A_i) = \beta_i$ is the actual quality of A_i delivered to the consumer. We define quality for a single attribute as:

Definition 1 (Quality) *Let A be a QoS attribute, $e(A)$ be the consumer's expectation on A and $d(A)$ be the provider's delivery on A. The quality for attribute A is defined as follows:*

$$QoS(A) = 1 - |d(A) - e(A)|$$

where $d(A)$ and $e(A)$ are normalised values in $[0, 1]$, with 0 representing the minimum level of quality and 1 the maximum.

Definition 1 implies that the quality of a delivered service ($d(A)$) will be perceived subjectively by different consumers based on their expectations ($e(A)$). For example, a service that delivered $d(A) = 0.2$ to a consumer with expectation of $e(A) = 0.2$ will be seen as delivering high quality of $QoS(A) = 1.0$. However, for a consumer whose

expectation was $e(A) = 0.9$, it will be seen as delivering low quality as $QoS(A) = 0.3$ (i.e. $QoS(A) = 1 - |0.2 - 0.9|$=0.3). So, while the data collected by the monitoring tools is objective, the actual quality observed is subjective, varying from one consumer to another.

We believe that capturing such subjectivity is essential to measuring QoS meaningfully. There are a number of factors that may determine what a preferred QoS level for a particular consumer should be. For example, the consumer may opt for a lower quality as he does not wish to pay for an expensive service, or he is limited by some constraints that any additional increase in quality would not benefit him, e.g. using an HD movie service on a screen with limited resolution.

Generally speaking, there are four possible perceptions of delivered quality by a consumer, as depicted in Figure 3.2.

For some attributes, it is reasonable to think that consumers would like their delivered values to be minimum or maximum. For example, for `availability` and `reliability`, consumers may like them to be as high as possible, and for `jitter` and `error rate`, as low as possible. However, due to price considerations, consumers may choose not to specify a minimum or maximum expectation on such attributes. This is explained in Figures 3.2(a) and 3.2(b), where the vertical line represents consumer expectation. In this case, if the actual delivered value is higher (in Figure 3.2(a)) or lower (in Figure 3.2(b)) than the expected value, it is unlikely that the consumer will be dissatisfied, i.e. the perceived quality will most likely increase too.

The situation is rather different in the cases described in Figures 3.2(c) and 3.2(d). Here, the minumum and the maximum values are not desirable. For example, a screen with a limited resolution will not benefit from a HD movies service. In such situations, the highest possible quality may be achieved with a particular delivered level of service leading to a flattening effect on the curve at a threshold point in Figure 3.2(c). Even worse, in some cases an increase in delivered level of service may cause a decrease in quality as perceived by the consumer. For example, an increase in the `frame rate`

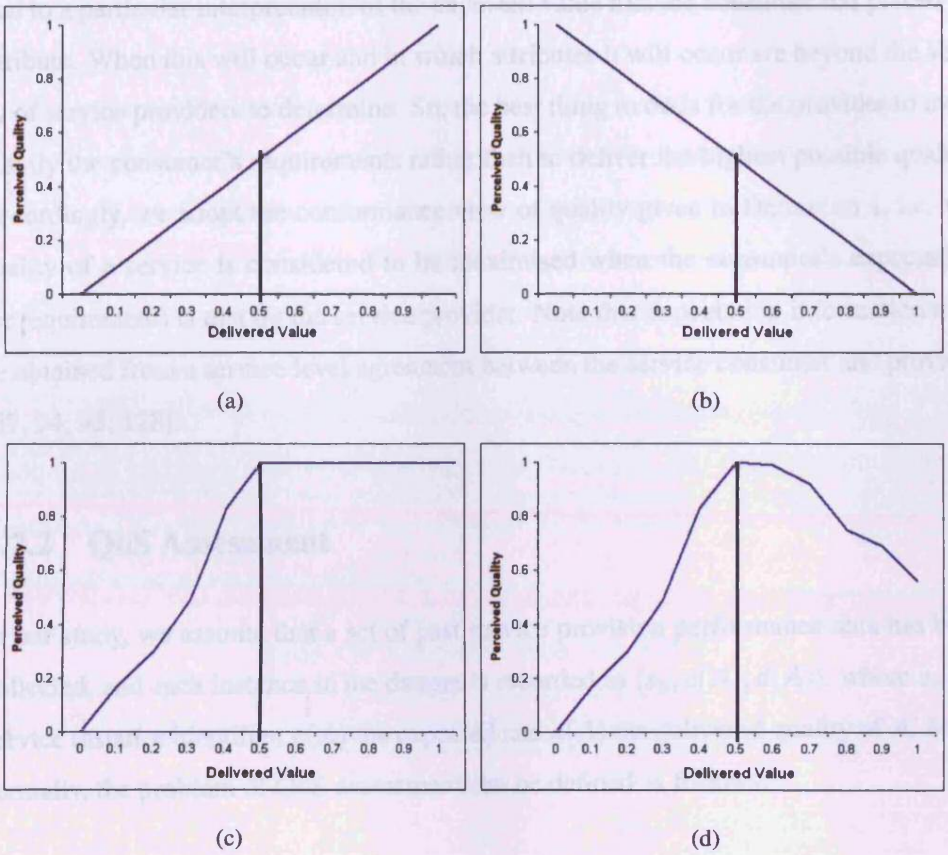


Figure 3.2: Perceived vs delivered quality

of a streaming movie service to a mobile phone may increase the perceived quality initially, but further increase in `frame rate` to a level that requires a level of buffering and memory use above the capability of the mobile phone may lead to increased `jitter` in the presentation of the video to the consumer, and hence a reduction in quality. In such a situation, the desired quality is achieved at a certain point and this is followed by a decline in quality for any further enhancement in the attribute, as shown in Figure 3.2(d).

From the above discussion it is clear that, in general, universal preferences cannot be assumed for all attributes and for all consumers. The specific context of a consumer may

lead to a particular interpretation of the expected value that the consumer has placed on attribute. When this will occur and in which attributes it will occur are beyond the ability of service providers to determine. So, the best thing to do is for the provider to meet exactly the consumer's requirements rather than to deliver the highest possible quality. Accordingly, we adopt the conformance view of quality given in Definition 1, i.e. the quality of a service is considered to be maximised when the consumer's expectation (or requirement) is met by the service provider. Note that expectation information may be obtained from a service level agreement between the service consumer and provider [67, 94, 95, 128].

3.2.2 QoS Assessment

In our study, we assume that a set of past service provision performance data has been collected, and each instance in the dataset is recorded as $\langle s_k, e(A), d(A)\rangle$, where s_k is a service instance identifier, $e(A)$ the expected and $d(A)$ the delivered quality of A. More formally, the problem of QoS assessment can be defined as follows:

Definition 2 (QoS Assessment) *Given a consumer quality requirement γ, the content of a QoS database, and a service S, determine the quality level that is likely to be delivered by S and the reliability of this prediction.*

From Definition 2, it can be seen that the output of a QoS assessment consists of two values: *prediction* and $confidence$. The *prediction* value falls in range [0..1] and indicates the level of service that the QoS assessment method believes would most likely be delivered to the consumer from a given service. It is important to emphasise that the *prediction* refers to a likely delivered level of service, rather than a likely quality of service. $Confidence$, on the other hand, represents the level of certainty with which the QoS assessment method is making the *prediction*. It also falls in range [0..1] where 0 indicates no confidence, and 1 complete confidence.

3.3 Assessing of Multiple Attributes

In this section, we discuss the limitations of current approaches to assessing QoS over multiple attributes. We then suggest how such limitations may be overcome. From our review in Chapter 2, there are two approaches to QoS assessment using monitored QoS data, non-selective and selective. We analyse two QoS assessment methods: Averaging All (AA) [22, 81] and Multiple Quality Space Mapping (MQSM) [127], as the representative methods of non-selective and selective approaches, respectively. To explain the issues associated to QoS assessment over multiple attributes, we use Table 1.1 introduced in Section 1.1 which is reproduced as Table 3.1. For ease of comparison, in Table 3.1 $d(A_1)$ refers to the delivered level by $S1$ for `access delay`, and $d(A_2)$ to the delivered level for `throughput`.

Table 3.1: Monitored QoS Data for $S1$

TID	$d(A_1)$	$d(A_2)$
t001	0.35	0.41
t002	0.67	0.72
t003	0.37	0.38
t004	0.71	0.83
t005	0.31	0.47
t006	0.65	0.87

3.3.1 Averaging All

To calculate QoS for $S1$, the simplest method is to average all the delivered service instances observed for each attribute of $S1$ first, and then average the aggregated values across attributes [22, 81]. This method was already introduced in Section 1.1, but is

described here again for ease of reference. That is, we calculate QoS as follows:

$$QoS(S) = \begin{cases} \sum_{j=1}^{m}(w_j \times \sum_{i=1}^{n} \frac{\beta_{ij}}{n}) & \text{if } n > 0 \\ default & n = 0 \end{cases} \tag{3.1}$$

where m is the number of attributes, n the number observed instances of S in the database, β_{ij} the observed delivered service value for A_j in the i-th instance, and w_j a weight indicating the significance of each attribute in aggregation. The closer the calculated $QoS(S)$ is to what is requested by a consumer, the higher the quality of S is considered to be for that consumer. When no previous provision instance has been observed, a default value will be returned.

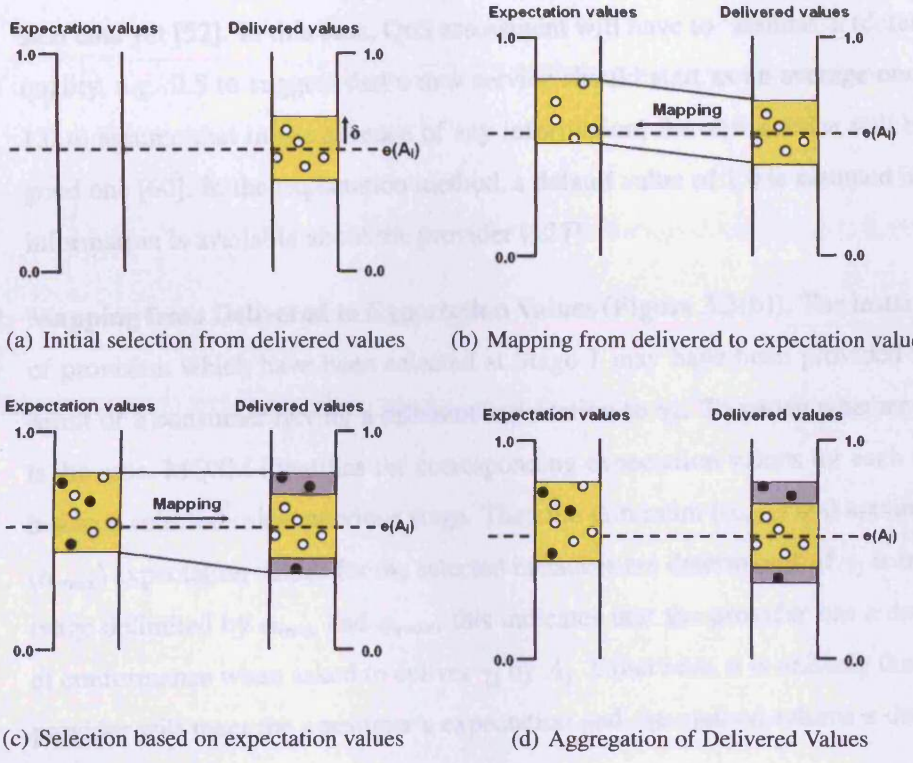
To apply this method to Table 3.1, we first calculate the quality level for A_1 and A_2 individually by averaging the observed data, i.e. $Avg(d(A_1)) = 0.51$ and $Avg(d(A_2) = 0.61$, and then the two averages are aggregated to give $QoS(S1) = 0.5 \times 0.51 + 0.5 \times 0.61 = 0.56$, assuming that the two attributes are equally important. By adopting quality as conformance (i.e. the degree to which a service provider meets (or conforms) to a particular level of service), this service is considered to offer good quality to a consumer who expects 0.55, since it delivered a level of service which is very close to what the consumer has asked for.

This method works fine if we assume that any level of service may be offered by each of the attributes. When this is not the case, for example, when A_1 is actually offered at two distinct levels at around 0.3 and 0.7 , then $QoS(A_1) = 0.51$ is unrealistic to get and the overall prediction of $QoS(S1) = 0.56$ is unlikely to materialise in practice.

3.3.2 Using Expectations

In [127] a multiple quality-space mapping (MQSM) method is proposed to provide a more accurate assessment when a service delivers multiple levels of quality. MQSM

also processes each attribute separately, but attempts to identify possible quality packages within a single attribute. To do so, the method goes through four main stages as shown in Figure 3.3.



(a) Initial selection from delivered values

(b) Mapping from delivered to expectation values

(c) Selection based on expectation values

(d) Aggregation of Delivered Values

Figure 3.3: A four-stage multiple quality space mapping method [127]

1. **Initial Selection from Delivered Values (Figure 3.3(a))**. In order to ascertain whether it is possible for a service provider to meet the consumer's expectation, MQSM attempts to find past service instances in which delivered values are similar to what the consumer expects. More specifically, service instances whose delivered values satisfy $|\beta_{ij} - \gamma_j| \leq \delta$ are selected from the database, where β_{ij} is the delivered value by A_j in i-th instance, γ_j is the consumer's expectation on A_j and δ denotes a bound intended to capture similar values. If any similar instances are found, then this is an indication that the service is capable (or has been ca-

pable) of delivering a service level that is desired by the consumer. If no data instances matched the consumer's expectation, a default value will be returned. This is similar to the cold start problem, which refers to the situation when a new service provider has just entered the system and has no historical service provision data yet [52]. In this case, QoS assessment will have to 'assume' a (default) quality, e.g. 0.5 to suggest that a new service should start as an average one or 1.0 to assume that in the absence of any information, the new service will be a good one [60]. In the expectation method, a default value of 1.0 is assumed if no information is available about the provider [127].

2. **Mapping from Delivered to Expectation Values (Figure 3.3(b)).** The instances of provision which have been selected at Stage 1 may have been provided as a result of a consumer having a different expectation to γ_j. To verify whether this is the case, MQSM identifies the corresponding expectation values for each data instance selected in the previous stage. Then the minimum (α_{min}) and maximum (α_{max}) expectation values for the selected instances are determined. If γ_j is in the range delimited by α_{min} and α_{max}, this indicates that the provider has a degree of conformance when asked to deliver γ_j by A_j. Otherwise, it is unlikely that the provider will meet the consumer's expectation and the method returns a default value.

3. **Selection based on Expectation Values (Figure 3.3(c)).** The provider may not consistently deliver what he has been asked to deliver. That is, for other instances with consumer expectation in the range $[\alpha_{min},\alpha_{max})]$ identified in Stage 2, the provider may also have delivered values in the past that are far away from the consumer's expectation (γ_j). To verify whether this is the case, MQSM visits the database again to retrieve a new set of instances whose expectation values satisfy $\alpha_{min} \leq \alpha_{ij} \leq \alpha_{max}$, where α_{ij} denotes the expectation value for A_j in the i-th instance. Note that the subset of data instances identified in this stage may not be the same as those identified in Stage 1.

4. **Aggregation of Delivered Values (Figure 3.3(d)).** The data instances identified in Stage 3 are aggregated to determine a likely level of service that may be delivered by the provider. That is, MQSM averages the corresponding β_{ij} values in this set of instances to obtain a prediction for A_j.

To illustrate how this method works, we expand Table 3.1 with expectation values and new data as shown in Table 3.2. Suppose that a consumer has the following expectation: $\gamma_1 = 0.4$ and $\gamma_2 = 0.9$ and we are asked to assess how likely $S1$ will meet this expectation. Assuming $\delta = 0.1$, MQSM first selects instances from Table 3.2 for A_1 based on $|d(A_1) - 0.4| \leq 0.1$ which gives us $\{t001, t003, t005\}$. From this set, we have $\alpha_{min} = 0.27$ and $\alpha_{max} = 0.41$. We then retrieve a new set of instances based on $0.27 \leq e(A_1) \leq 0.41$ which gives us the same data set. Finally, we aggregate the delivered values in this set to find $QoS(A_1) = 0.34$. Similarly, MQSM selects instances for A_2 based on $|d(A_2) - 0.9| \leq 0.1$ which gives us $\{t004, t006\}$. For this set, we have $\alpha_{min} = 0.80$ and $\alpha_{max} = 0.87$. We then retrieve a new set of instances based on $0.80 \leq e(A_2) \leq 0.87$ which gives us $\{t002, t004, t006\}$. This adds instance $t002$ to those already identified. Finally, we aggregate the delivered values in this set to find $QoS(A_2) = 0.81$. Assuming that the two attributes are equally important, the overall prediction for $S1$ is computed to be $QoS(S1) = 0.5 \times 0.34 + 0.5 \times 0.81 = 0.58$.

Table 3.2: Historical QoS Data for $S1$ with Expectations

TID	$\langle e(A_1), d(A_1)\rangle$	$\langle e(A_2), d(A_2)\rangle$
t001	$\langle 0.27, 0.35\rangle$	$\langle 0.35, 0.41\rangle$
t002	$\langle 0.77, 0.67\rangle$	$\langle 0.85, 0.72\rangle$
t003	$\langle 0.41, 0.37\rangle$	$\langle 0.21, 0.38\rangle$
t004	$\langle 0.65, 0.71\rangle$	$\langle 0.80, 0.83\rangle$
t005	$\langle 0.36, 0.31\rangle$	$\langle 0.28, 0.47\rangle$
t006	$\langle 0.69, 0.65\rangle$	$\langle 0.87, 0.87\rangle$

If we consider the accuracy of the assessment of a single attribute, MQSM performs better than the Averaging-All method as both $QoS(A_1) = 0.34$ and $QoS(A_2) = 0.81$ are clearly meaningful. However, if we assume that quality is offered with some "packaging" across multiple attributes, e.g. one package offers around 0.3 for A_1 and around 0.4 for A_2, and another offers around 0.7 for A_1 and around 0.8 for A_2, then the predicted *combination* $QoS(A_1) = 0.34$ and $QoS(A_2) = 0.81$ is clearly unattainable and $QoS(S1) = 0.58$ is misleading.

3.3.3 Synchronous Extension (SE)

To deal with multiple attributes correctly, we can apply a simple "adjustment" to the MQSM method described in Section 3.3.2. Instead of selecting and aggregating $d(A_1)$ and $d(A_2)$ separately first in Table 3.2 and then combining $QoS(A_1)$ and $QoS(A_2)$ into a single verdict, we can go through the four stages of MQSM to select individual instances based on both γ_1 and γ_2, and then aggregate the qualified $d(A_1)$ and $d(A_2)$ that result from Stage 3 into a single prediction for $S1$. That is, we perform:

$$QoS(S) = \begin{cases} \sum_{i=1}^{k}(\sum_{j=1}^{m} w_j \times \beta'_{ij})/k & \text{if } k > 0 \\ default & k = 0 \end{cases} \tag{3.2}$$

where k is the number of instances that satisfy $|\beta'_{i1} - \gamma_1| \leq \delta, |\beta'_{i2} - \gamma_2| \leq \delta, \ldots, |\beta'_{im} - \gamma_m| \leq \delta$ simultaneously. Applying this to our running example, it is easy to verify that no instances satisfy $|d(A_1) - 0.4| \leq 0.1$ and $|d(A_2) - 0.9| \leq 0.1$ simultaneously, hence a default result will be reported. This verdict is clearly more meaningful and is a more accurate statement of what $S1$ is able to deliver: the required level of service $\gamma_1 = 0.4$ and $\gamma_2 = 0.9$ is not in fact attainable from $S1$.

The problem with this simple adjustment is that it implicitly assumes that QoS data for the multiple attributes involved is synchronously collected, as we require $|\beta'_{i1} - \gamma_1| \leq$

$\delta, |\beta'_{i2} - \gamma_2| \leq \delta, \ldots, |\beta'_{im} - \gamma_m| \leq \delta$ to be simultaneously satisfied. This is a rather restrictive assumption which is unlikely to be held in practice, because multiple QoS attributes are more likely to be monitored independently at different time points and at different rates [148, 66, 130]. If we allow QoS data to be collected asynchronously, then we can anticipate that the number of instances in the database that satisfy our required condition will be significantly reduced, particularly when we have a large number of attributes. This in turn can seriously reduce the confidence of assessment. To illustrate this, assume that the data given in Table 3.2 for $S1$ has in fact been collected as in Table 3.3, i.e. some monitored data has not been collected at the same time.

Table 3.3: Asynchronously Collected QoS Data for $S1$

TID	$\langle e(A_1), d(A_1)\rangle$	$\langle e(A_2), d(A_2)\rangle$
t001	$\langle 0.27, 0.35\rangle$	
t002		$\langle 0.35, 0.41\rangle$
t003	$\langle 0.77, 0.67\rangle$	$\langle 0.85, 0.72\rangle$
t004	$\langle 0.41, 0.37\rangle$	
t005		$\langle 0.21, 0.38\rangle$
t006	$\langle 0.65, 0.71\rangle$	
t007		$\langle 0.80, 0.83\rangle$
t008	$\langle 0.36, 0.31\rangle$	$\langle 0.28, 0.47\rangle$
t009	$\langle 0.69, 0.65\rangle$	
t010		$\langle 0.87, 0.87\rangle$

Suppose that $S1$ is to be assessed for a consumer request: $\gamma_1 = 0.7$ and $\gamma_2 = 0.8$. Assuming $\delta = 0.1$, clearly in this case only `t003` will be selected by the SE method in assessment. Hence, SE gives quality for A_1 and A_2 as $QoS(A_1) = 0.67$ and $QoS(A_2) = 0.72$. While this still gives a correct assessment, intuitively the confidence of this assessment will be low as only a very small fraction of the data is actually used in assessment. We will study this further in Chapter 4.

3.4 Summary

In this chapter, we introduced a conceptual model for QoS assessment and used it to analyse the problems associated with assessing QoS over multiple attributes. We considered a QoS assessment process to consist of four main tasks: data collection, data selection, data aggregation and service ranking. We then gave our definition of quality as the difference between what is requested by a consumer and what is actually delivered to the consumer.

The focus of this chapter is on how to assess QoS over multiple attributes more effectively. We discussed two representative QoS assessment methods, Averaging-All and MQSM. To assess QoS over multiple attributes, these two methods largely assume that the data in each attribute may be selected and aggregated individually. We analysed the limitation of such individual assessment using the example introduced in Chapter 1. Generally, the Averaging All method works fine if it is assumed that any level of quality may be offered by each of the attributes regardless of user expectations. When this is not the case, for example when a service offers multiple levels of quality to consumers based on their expectations, the method gives an incorrect assessment.

MQSM, on the other hand, is able to identify different levels of quality, but is limited to considering possible levels within a single attribute only. Although MQSM method can improve the accuracy of QoS assessment over a single attribute, it would not work well with multiple attributes. To deal with multiple attributes more effectively, we applied a simple "adjustment" to the MQSM method. The synchronous Extension (SE) method selects and aggregates the data of multiple attributes collectively. In doing so, the method can improve the accuracy of QoS assessment over multiple attributes. However, this simple extension implicitly assumes that QoS data over multiple attributes is synchronously collected. When this is not the case, we can anticipate that the number of instances used by SE in assessment will be significantly reduced. This in turn can seriously reduce the confidence of assessment.

To improve accuracy and confidence when assessing QoS over multiple attributes, a new method is needed. In the following chapters, we propose a new method to address these.

Chapter 4

Modelling Confidence for QoS Assessment

This chapter develops a confidence model for QoS assessment. This model is used to deal with uncertainty surrounding a QoS assessment. The chapter is organised as follows. In Section 4.1, we review some sources of uncertainty surrounding the QoS verdicts produced by QoS assessment method. We then explain how decision and probability theory may help to deal with uncertainty in QoS assessment. In Section 4.2, we discuss various approaches in the literature to establishing confidence measures and then describe our proposed confidence model. Following that, we highlight the limitations of existing QoS assessment methods using our proposed confidence measure. Finally, a summary of the chapter is provided in Section 4.3.

4.1 Dealing with Uncertainty

In an open and dynamic environment, any techniques and tools that attempt to help assess QoS will have to deal with some uncertainty surrounding the QoS verdicts they give. This is because in such an environment, service providers' behaviours cannot always be expected to be stable over time. That is, their performance may fluctuate due to a range of factors, e.g. network congestion, resource constraints or simply lack of good quality management [120, 69, 63, 147]. It is easy to see that such variation in service

delivery performance will be reflected in the collected QoS data, and a QoS verdict based on a set of data that has a large variation is likely to be unreliable. Moreover, when QoS assessment is conducted on a small set of data (e.g. when assessing a new service or only considering an asynchronous subset of the data in assessing multiple attributes), then intuitively the verdict may not be reliable either. Thus, it is essential that we attach a confidence value to a QoS assessment, so that the level of certainty surrounding the verdict can be indicated to the assessment requester, and can be taken into account when he selects the preferred service to use.

To deal with uncertainty in our QoS assessment, we use Decision theory [133]. The key idea underlying decision theory is quite simple: people always choose actions that move them towards situations, or states, that they prefer. Given a set of alternatives and a set of consequences following each alternative, decision theory models the relationship between the two sets and offers a conceptually simple procedure for choosing among alternatives. In the example introduced in Section 1.1, Alice wants a web hosting service that can serve 800 requests per second, and she needs to choose between four web hosting services ($S1 - S4$). Suppose that $S1$ is able to deliver 200 requests per second, whereas $S2$ can deliver 800 requests per second. Then, Alice would prefer $S2$ to $S1$[1], as taking this action will leave her in a better 'state', i.e. receiving a throughput of 800 requests per second.

To capture what is a preferred state to a consumer, consumer expectations on quality of services are encoded in a *utility* function that maps the set of situations the consumer may find himself in to a set of real numbers representing the value of each situation to that consumer. This is explained in the next section.

[1]We assume that Alice is only interested in getting the required throughput and has no other requirements, e.g. a certain price range or a requirement on response time.

4.1.1 Calculation of the Expected Utility of the Consumer

The consumer's decision-making is based on the utility they expect to obtain from a provider. To move to the state that they most prefer, a consumer always chooses actions that maximise his utility. This is done such that, if one state is preferred to another, then the utility value returned for the preferred state will be higher than that for the other. Similarly, if two states are equally preferred, then they should have equal utility. Formally, if Θ is the set of all states the consumer can reach through his own actions, and the utility of each $\theta \in \Theta$ is given by the utility function U(θ), then the consumer should choose to act so as to arrive in a state $\bar{\theta}$, such that:

$$\forall \theta \in \Theta, U(\theta) \leq U(\bar{\theta}) \tag{4.1}$$

Unfortunately, it is not always feasible to know how to act in order to arrive in state $\bar{\theta}$. Typically, if a consumer has to choose between competing providers (such as $S1 - S4$ in Alice's example) he will not know for certain which provider will act most in his favour (i.e. be closest to meeting his expectation). To deal with this problem, decision theory draws upon probability theory [92] which provides a reasonable method to address this uncertainty. That is, the consumer should act to maximise expected utility, i.e. the utility they expect to obtain from a service if the provider is true to what he has promised, and the likelihood that the provider is able to do this. More formally, expected utility function for a single attribute can be defined as follows:

Definition 3 (Expected Utility Function for Single Attribute) *The consumer C_j's expected utility function for attribute A_k of service S_i is defined as:*

$$EU_k^j(\theta) = \int_{\theta \in \Theta} p(\theta|a)U(\theta)d\theta$$

where θ is the state that C_j will move to w.r.t. attribute k and $p(\theta|a)$ is the probability of θ given that the consumer takes action a, i.e. selects to use service S_i.

The expected utility of the consumer based on all QoS attributes is as follows:

Definition 4 (Expected Utility Function for Multiple Attributes) *The consumer C_j's expected utility function for all attributes of service S_i is defined as:*

$$EU^j(\theta) = \sum_{k=1,\ldots,m} EU^j_k(\theta) = \sum_{k=1,\ldots,m} \int_{\theta\in\Theta} p(\theta|a)U(\theta)d\theta$$

where θ is the state that C_j will move to w.r.t. all attributes of service S_i and $p(\theta|a)$ is the probability of θ given that the consumer chooses action a, i.e. selects to use service S_i.

To derive a representation of consumer's utility $U(\theta)$ in Definitions 3 and 4, we need to understand consumers' expectations or requirements for QoS. Conformance has been introduced in Chapter 3 as the means by which consumers judge the quality of a service which is maximised when the consumer's expectation is met by the service provider. So, the utility ($U(\theta)$) obtained from a particular service will correspond to the degree of conformance to the consumer's expectation.

The presence of probability in the definition of expected utility means that decision theory in the face of uncertainty enters the realm of statistics. That is, to determine the expected utility of an action, we must determine the probability distribution of the possible states an action will result in. How we determine this distribution depends on the nature of the problem at hand. The decision of choosing between n candidate services based on information from a QoS assessment method can be viewed as choosing a service that exhibits the best expected utility. In such a decision, the factor that potentially affects the utility of each choice is the accuracy of prediction by a QoS assessment method regarding the performance of the candidate service. Calculating the expected utility as in Definition 4 for each candidate service S_i will result in an expected utility value for each choice. Rationally, a consumer will choose the candidate service that

corresponds to the action offering the highest expected utility value. More specifically,

$$S_{opt} = arg\ max\ EU(S_i) \tag{4.2}$$

It should be noted that the decision made by the consumer based on Equation 4.2 considers QoS only. There are, of course, many other considerations that might be taken into account by the consumer in choosing a service, such as the cost or reputation of the service provider. To accommodate these factors in the consumer's decision-making, the above expected utility may be combined with other sources of information in making an overall decision. This thesis, however, does not consider the details of such issues.

4.2 Modelling Confidence

In this section, we show how decision theory in conjunction with probability theory can be applied to our specific problem, where the consumer needs to choose whether or not to use a potential service provider. We first survey the existing confidence models to examine if they can be adapted to addressing our problem.

4.2.1 Current Confidence Models

There are many confidence models proposed in the literature, especially in the area of trust and reputation systems. The main aim of these models is to deal with uncertainty surrounding the behaviour of consumers when producing their ratings. Sabater et al. [117] introduced two measures to calculate the reliability of trust value; the number of ratings and their deviation. The authors in [54] and [68] concur with the assertion of using these two measures to calculate trust value's reliability. The work in [96] uses the Chernoff Bound to determine the minimum sample size required to achieve a certain level of confidence. This approach is also used by the author in [160] to compute if the experience (measured by sample size) of an agent is sufficient enough

(i.e. reliable enough) to reason about the likely behaviour of other agents. Overall, studies have noted the importance of considering data characteristics (measured by size and deviation) in deriving a confidence value.

Unfortunately, the existing confidence models cannot be directly applied in our work for two reasons. First, they are more for discrete and subjective data (user ratings), while the data we are dealing with is continuous and objective (monitored data). Second, the reliability verdict is derived "statically" and ignores consumer expectations, which is important in calculating our utility.

Adopting the conformance view of quality in our work imposes more complexity in confidence computation. This is because what is assumed to be good enough for one consumer may not be for another. For example, a hosting web service that delivers 400 requests per second to Alice who was promised to receive 800 requests per second will be judged to be unsuccessful. However, this does not mean that the same delivered value (400 requests/second) will be judged by other consumers in a similar way. For example, the same delivered level of service would be enough and satisfactory for someone who expected 390 requests per second. So, in general, in contrast to the existing confidence methods, to determine whether a given data instance is reliable or not we must "dynamically" derive a verdict with respect to the required quality. In the next section, we propose a confidence model that can handle this issue adequately and deal with uncertainty surrounding a QoS prediction.

4.2.2 Proposed Confidence Model

When a QoS verdict is derived for a given service using its past performance data, it is essential that we are able to establish its reliability or the confidence level to be placed on the verdict. In our context, the confidence is used to deal with uncertainty surrounding the reliability of a QoS prediction. More specifically, it expresses how confident QoS assessment method is in producing the assessment result given the data

used in assessment. It is an important measure because a consumer's decision about which service to choose may depend on the service's behaviour over time, e.g. its stability to deliver the required level of service. So a good QoS assessment should give an indication on the reliability of its assessment result, so that consumers will be able to make a more informed decision in selecting their preferred services.

In our confidence model, we use the two measures suggested by the models reviewed in Section 4.2.1 to calculate the confidence of a prediction for a single attribute: data size (Rel_ω) and data deviation (Rel_θ). These two measures are important in helping consumers select their preferred services. While the former indicates how strongly the prediction derived by the QoS assessment method is supported by the dataset, the latter indicates the service's consistency in delivering that prediction. Applying the two measures in our work, however, is not straightforward due to the adopt on the conformance view of quality as explained in Section4.2.1. Thus, we have made some modification to the two measures in order to make them well suited for our context. This will be discussed further in the next sections. Figure 4.1 shows how the proposed confidence model is integrated into the QoS assessment process introduced in Section 3.1.

Data Size Measure (Rel_ω)

This measure is based on the number of data items used in assessment. Each data item used in assessment provides an independent piece of evidence about the quality that a service has offered in the past. So intuitively, the more evidence we have, the higher confidence we should have for the assessment. More formally, this is captured in Equation 4.3 which states that as the number of data items grows, the degree of reliability increases until it reaches a defined threshold denoted by m:

$$Rel_\omega = \begin{cases} \frac{\bar{n}}{m} & \text{when } \bar{n} < m \\ 1 & \bar{n} \geq m \end{cases} \tag{4.3}$$

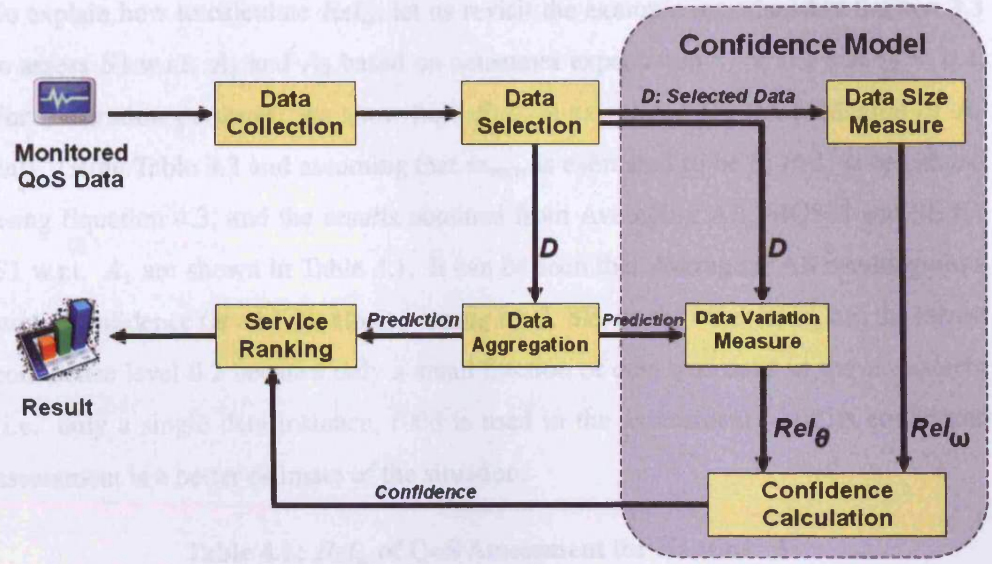


Figure 4.1: Integration of confidence model into QoS assessment process

where $\bar{n}$ is the number of data items selected and used in assessment. So Rel_ω increases from 0 to 1 as the number of selected data items $\bar{n}$ increases from 0 to m, and stays at 1 when $\bar{n}$ exceeds m.

To determine the minimum number of data items (m_{min}) that is needed in order to achieve a certain level of confident about an assessment, we use the Chernoff Bound [96] to calculate m_{min}:

$$m_{min} = -\frac{1}{2\epsilon^2} ln \frac{1-\lambda}{2} \tag{4.4}$$

where ϵ is the maximal level of error that can be accepted by the consumer, and λ is the required confidence level. So the larger the λ and the smaller the ϵ are, the larger m_{min} is required. For example, if we set $\lambda = 0.99$ and $\epsilon = 0.1$, then the minimum number of data items needed is $m_{min} = 1060$. While this suggests that we should use as many recorded QoS data items as possible in order to have confidence in assessment, care must be taken, as we have explained in Section 3.3.1, that we do not use data that may give us a misleading verdict.

To explain how to calculate Rel_ω, let us revisit the example introduced in Section 3.3 to assess $S1$ w.r.t. A_1 and A_2 based on consumer expectation $\gamma_1 = 0.4$ and $\gamma_2 = 0.4$. For illustration purposes, we show how Rel_ω is calculated for the prediction of A_1 only. Using Table 3.3 and assuming that m_{min} is estimated to be 5, Rel_ω is calculated using Equation 4.3, and the results obtained from Averaging All, MQSM and SE for $S1$ w.r.t. A_1 are shown in Table 4.1. It can be seen that Averaging All results gives a strong confidence (level 1.0), albeit a wrong level. SE, on the other hand, has the lowest confidence level 0.2 because only a small fraction of data was used in the assessment (i.e. only a single data instance, $t008$ is used in the assessment), but its confidence assessment is a better estimate of the situation.

Table 4.1: Rel_ω of QoS Assessment for $S1$ w.r.t. A_1

Assessment Method	*prediction*	$\bar{n}$	Rel_ω
Averaging All	0.51	6	1.0
MQSM	0.34	3	0.6
SE	0.31	1	0.2

The Rel_ω measure is a quantitative metric that seeks to use as much data as possible to support QoS assessment prediction. This measure, however, does not consider data variation which will be considered by the Rel_θ measure in the next section.

Data Deviation Measure (Rel_θ)

This measure is based on the variation within the data used in assessment. When selected QoS data are aggregated into a single verdict, it is important to take data variation into account. This is because different data distributions may average to the same mean, yet have a greater variation in data (i.e. have a more fluctuate service delivery). This should intuitively suggest that the derived mean is a less reliable verdict.

To capture variation within the data used in assessment, we view service delivery as a set of Bernoulli trials: *successful delivery* (delivered the required service level) or *unsuccessful delivery* (did not deliver the required level), and then model it as Beta distributions

$$Rel_\theta = \frac{\alpha}{\alpha + \beta} \tag{4.5}$$

where $\alpha = r + 1$ and $\beta = s + 1$, and r is the observed number of successful deliveries and s the unsuccessful ones. The ratio of α and β will determine where in the interval [0,1] the distribution peaks, and a high α will cause the distribution mode to occur close to 1.

The idea of adding 1 each to r and s (and thus 2 to r+s) follows Laplace's *rule of succession* for applying probability to inductive reasoning [114]. This rule in essence reflects the assumption of an equi-probable prior, which is commonly adopted in probabilistic reasoning. That is, when we have no information about a service initially, (i.e. r=0 and s=0), we have Rel_θ=$\frac{1}{1+1}$ =0.5, suggesting that a successful and unsuccessful delivery of the service by the provider is equally likely.

To determine whether a past delivery on a single QoS attribute was a successful one or not, we use the following:

$$x(A_i) = \begin{cases} 1 & d(A_i) \in [p_i - \epsilon_\theta, p_i + \epsilon_\theta] \\ 0 & \text{Otherwise} \end{cases} \tag{4.6}$$

where $d(A_i)$ is a delivered quality on A_i in the selected dataset D and p_i is the verdict given by the assessment on attribute A_i (i.e. the aggregated value which is derived by a simple average in our context). If $d(A_i)$ is within a specified ϵ_θ from p_i, then it is considered to be a successful delivery, denoted by $x(A_i) = 1$. Otherwise, it is unsuccessful, denoted by $x(A_i) = 0$. We call the range, i.e. $[p_i - \epsilon_\theta, p_i + \epsilon_\theta]$, that used to determine if a past delivery is a successful one or not a confidence range. Accordingly, r and s are:

$$r = \sum_{x_i \in D} (x_i = 1) \;\; and \;\; s = \sum_{x_j \in D} (x_j = 0) \tag{4.7}$$

Clearly, choosing different values for ϵ_θ can have a direct impact on Rel_θ. For the same data size, a large ϵ_θ can result in a more confident QoS verdict, because it will result in more α's (successful deliveries) and consequently increase Rel_θ. Thus, a large ϵ_θ may be chosen when a consumer is willing to accept a more fluctuated delivery of a service from a service provider.

Note that in contrast to other models in the literature [59, 96, 135, 160], whether a past delivery is a successful one or not is determined dynamically in our model. This is because p_i is computed based on user expectation $e(A_i)$, so whether $d(A_i)$ is a successful delivery or not is affected by $e(A_i)$. For example, if S is delivered with a quality of 0.5 on attribute A_i, then it is deemed as a successful one for a consumer who requested 0.5 on A_i (i.e. $e(A_i) = 0.5$), but not for the one who requested 0.9, if we set $\epsilon_\theta = 0.1$. A small range $[p_i - \epsilon_\theta, p_i + \epsilon_\theta]$ is used here as it is reasonable to assume that consumers will be satisfied by a delivery that is close enough to the requested quality level.

To explain how Rel_θ may be calculated, let us revisit the example in Section 3.3 to assess $S1$ w.r.t. A_1 and A_2 using Averaging All, MQSM and SE based on consumer expectation $\gamma_1 = 0.4$ and $\gamma_2 = 0.4$. For illustration purposes, we show how the Rel_θ for the prediction of A_1 is calculated only. Given the data selected by Averaging All, D=$\{0.35, 0.67, 0.37, 0.71, 0.31, 0.65\}$ (all values of the $d(A_1)$ column in Table 3.3) and its prediction $p_1 = 0.51$ and assuming that $\epsilon_\theta = 0.05$, whether each $d(A_i)$ is a successful delivered value or not is calculated using Equation 4.6. It is easy to see that all delivered values are classified as unsuccessful because they are out of the confidence range delimited by p_1 and ϵ_θ, $[0.46, 0.56]$. For MQSM, only three data items are selected as relevant to $e(A_1)$, D=$\{0.35, 0.37, 0.31\}$. All these data items are classified as successful because they fall in the confidence range $[0.29, 0.39]$. Similarly, the single data item selected by SE, $D = \{0.31\}$, is classified as successful because it falls in the confidence range $[0.26, 0.36]$. Using Equation 4.5, Rel_θ is calculated and the result is shown in Table 4.2.

Table 4.2: Rel_θ of QoS Assessment for $S1$ w.r.t. A_1

Assessment Method	α	β	Rel_θ
Averaging All	1	7	0.125
MQSM	4	1	0.80
SE	2	1	0.67

Overall Confidence Value

Intuitively, the Rel_ω measure indicates how strongly the mean derived by the QoS assessment method is supported by the dataset and the Rel_θ measure indicates the service's consistency in delivering that mean. Based on these two measures, we calculate the overall confidence as follows:

$$Confidence = Rel_\omega \times Rel_\theta \tag{4.8}$$

Table 4.3 shows the overall confidence that Averaging All, MQSM and SE have in their prediction when assessing $S1$ w.r.t. A_1 based on consumer expectation $\gamma_1 = 0.4$.

Table 4.3: Confidence of QoS Assessment Methods on assessing $S1$ w.r.t. $\gamma_1 = 0.4$

Assessment Method	*prediction*	Rel_ω	Rel_θ	Confidence
Averaging All	0.51	1.0	0.125	0.125
MQSM	0.34	0.6	0.80	0.48
SE	0.31	0.2	0.67	0.134

It is clear from Table 4.3 that the proposed confidence measure is able to identify cases where predictions by a QoS assessment method is unreliable. For example, the overall confidence for Averaging All's prediction was low because it did not adequately cope with the situation when a service delivered multiple levels of service. The Averaging All

method selected all available data as relevant to the requester's assessment, thus mistakenly mixed up different levels of QoS offerings to produce its prediction. Although Averaging All's prediction was well supported by the Rel_ω measure, it was discounted by the Rel_θ measure. The high confidence of MQSM's prediction provides further evidence for the preceding findings. While this method selected part of the data in the assessment and was supported less by Rel_ω (0.6), its prediction was more accurate (i.e. stick with one of the service levels delivered by $S1$). The confidence of SE's prediction, on the other hand, provided clear evidence of how QoS assessment may be affected by asynchronous data. While the prediction of SE was as accurate as MQSM, its overall confidence was low, similar to the confidence for Averaging All. This is because, in contrast to MQSM, the prediction by SE was less supported by Rel_ω. This contributed an overall low confidence for SE prediction, despite being an accurate one. Thus, the most important issue is to determine which QoS data should be used in assessment, to ensure that assessment accuracy and confidence are best balanced.

4.2.3 Discussion

To show how our proposed confidence measure may help QoS assessment, consider the following example. Suppose that we have two services $S1$ and $S2$, a consumer C who wishes to find a service that has a certain capability, and both $S1$ and $S2$ can offer the required capability. Suppose also that C requires a certain quality on A (i.e. $e(A) = \rho$, where A is an attribute of both $S1$ and $S2$), and QoS assessment indicates that both $S1$ and $S2$ can meet this requirement (i.e. their past performance data on A average to a similar mean). In this case, as both $S1$ and $S2$ offer similar qualities, our confidence calculation can be used as a further differentiator for service selection.

Assume that the distributions of the collected QoS data for $S1$ and $S2$ are shown in Table 4.4. The successful and unsuccessful deliveries (r and s) have been calculated according to the method given in Section 4.2.2, and both average to a similar mean.

Table 4.4: QoS Data Distribution for $S1$ and $S2$

Service	r	s
$S1$	150	100
$S2$	50	100

Using Table 4.4 and assuming that m_{min} is estimated to be 200, Rel_ω and Rel_θ are calculated using the equations given in Section 4.2.2, and the overall confidence of the assessment for $S1$ and $S2$ w.r.t. A is shown in Table 4.5. It is clear that the assessment for $S1$ is more reliable, so $S1$ should be considered as the best candidate for C.

Table 4.5: QoS Assessment with Confidence

Service	r	s	Rel_ω	Rel_θ	Confidence
S_1	150	100	1.0	0.599	0.599
S_2	50	100	0.75	0.336	0.252

The proceeding example illustrates the usefulness to consider confidence in QoS assessment. This is particularly important in dealing with multiple QoS attributes that are asynchronously monitored. While simple solutions exist to handle multiple attributes, e.g. by considering the synchronous subset only (Section 3.3.3), the amount of available data for assessment may be significantly reduced, particularly when the dimensionality is high. As we have shown in this section, this could mean that an unreliable verdict is delivered to a consumer. Motivated by this consideration, we consider how to handle asynchronous data in QoS assessment in the next chapter.

4.3 Summary

In this chapter, we have introduced a confidence model for QoS assessment that can be used to deal with uncertainty surrounding the reliability of a QoS prediction. The chapter started by indicating some sources of uncertainty that consumers may face in making a decision about which service to use. We then described how decision theory may help address such uncertainty. To support consumer selection of the most preferred service, a QoS assessment method returns a predicted level of service for each candidate service associated with confidence value. The prediction is used to evaluate the level of service that the consumer may receive from using each candidate, while the confidence value is used to determine the level of uncertainty surrounding the prediction.

To quantify confidence in QoS assessment, we have presented a probabilistic model that integrates two reliability measures: the number of QoS data items used in assessment and the variation of data in the dataset. In this model, the data size measure indicates how strongly the prediction derived by the QoS assessment method is supported by the dataset, and the data deviation measure indicates the service's consistency in delivering that prediction. Several examples have been used to show how the proposed model works and the usefulness of including confidence in QoS assessment. This is particularly important when dealing with multiple QoS attributes that are asynchronously monitored. In the next chapter, we consider how to handle asynchronous data in QoS assessment so that consumers can make a more rational decision in choosing their preferred services.

Chapter 5

Handling Asynchronous QoS Data

As we have shown in the previous chapter, when QoS data across multiple attributes are not synchronously collected, the method suggested in Section 3.3.3 could have substantially less data to use. This will affect the confidence of assessment, particularly when we have a large number of attributes. In this chapter, we propose a solution to this problem. We consider asynchronous data as a set of data containing "missing" values and employ a kNN based technique to estimate the missing ones. In doing so, we increase the amount of usable data. That is, we attempt to transform asynchronous data into a synchronous form. We also show how our proposed solution may be integrated into a QoS assessment method to provide more accurate and confident QoS assessment over multiple attributes.

The chapter is organised as follows. In Section 5.1 we show the importance of data preparation in data analysis to enhance the quality of data. Then, we define the problem of asynchronous data in Section 5.2. Various methods to predict missing data are described and discussed in Section 5.3. We propose our solution to handling asynchronous data in Section 5.4. We then show how our proposed solution may be integrated into a QoS assessment in Section 5.5. Finally, we summarise the chapter in Section 5.6.

5.1 Importance of Data Preparation

Using data collected by monitoring tools for QoS assessment can eliminate several problems associated with user ratings and advertisements (see Section 2.4.4). Since users do not play a direct role in data collection, the QoS assessment is free from user manipulation, making the assessment result more reflective of the actual level of service quality offered by the provider. However, some factors may affect the effectiveness of QoS assessment methods using monitored data in real world applications. Among them, the representation and quality of the collected data are of importance. In many computer science fields, such as pattern recognition, information retrieval, machine learning, data mining, and web intelligence [71, 109, 161, 159], one relies on data of good quality, and the collected data often needs to be pre-processed in order to enhance data quality. In our work, however, we do not consider such issues and simply assume that the collected data is of good quality and is a perfect representation of the real world.

However, despite our assumption on data quality, there is one data issue that must be considered in QoS assessment. For multiple attributes, it is more likely that the data on quality are collected asynchronously w.r.t. time and at different rates [148, 66, 130]. Thus, the methods that implicitly assume that QoS data is synchronously collected will have a small fraction of data to use in assessment, which will result in low confidence in assessment, as explained in Section 2.3. In this chapter, we propose a solution to address this problem. That is, we suggest preparing the QoS data before making assessment by transforming asynchronous data into a synchronous form. In doing so, we can increase the amount of usable data, thus enhance QoS assessment performance to produce a more reliable assessment and consequently better service ranking and selection.

5.2 Problem Definition and Formulation

Before describing the detail of the proposed approach, we give a definition of asynchronous data. Informally, a set of QoS data is not synchronously collected if at any given time, we may find that data has been collected for some, but not for other attributes of a service. More formally,

Definition 5 (Asynchronous QoS Data) *Given a service* $S(A_1, A_2, \ldots, A_m)$*, a set of QoS data* D *collected for* S *is* asynchronous *if there exists one instance* $\langle S, t_h, e(A_i), d(A_i)\rangle$ *in* D *such that there is at least one* $\langle S, t_h, e(A_j), d(A_j)\rangle$*,* $i \neq j$*, that cannot be found in* D*.*

Note that asynchronous data can result from different monitoring techniques and sampling rates that have been enforced on different attributes during service monitoring [5, 6, 95, 130]. The setting of these rates is normally based on some practical requirements or trade-offs [148], and is beyond the control of QoS assessment. For example, `response time` may need to be measured after each query, and a trade off must be considered between using a high sampling rate for fuller monitoring of a service's behaviour and the overhead caused by it. In general, therefore, we expect a set of QoS data collected from multiple attributes to be asynchronous.

In our work, we treat asynchronous data as a set of data containing "missing" values. That is, we see the set of collected data D for S as shown in Table 5.1, where $\perp$ in (i, j) indicates that no data has been collected at t_i for A_j. Our approach is to estimate (or impute) such missing values using other information available within D before applying QoS assessment algorithms to the data.

Definition 5 captures the notion of synchronicity generically, but does not specify how synchronicity may be determined. Unfortunately, it is not always straightforward to

Table 5.1: Asynchronously Collected QoS Data

time	A_1	A_2	A_3	$\cdots$	A_m
t_1	v_{11}	v_{12}	$\perp$	$\cdots$	v_{1m}
t_2	$\perp$	v_{22}	v_{23}	$\cdots$	$\perp$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
t_n	v_{n1}	v_{n2}	v_{n3}	$\cdots$	v_{nm}

determine if data instances over multiple attributes are synchronous or not. In Definition 5, synchronicity requires *exact* matching w.r.t. time between different data instances collected from different attributes, because it defines "matching" in term of an exact timepoint t_h. However, it may not be realistic to expect this to happen in real world applications. For example, if v_{11} was collected at 00:10:09 and v_{12} at 00:10:10 in Table 5.1, they will be considered to be asynchronous, although actually they are probably *close* enough to be considered synchronous. One possible approach to handling such cases is to relax the definition of synchronisation given in Definition 5 by allowing monitored QoS data to be synchronised within a "time window". This method is commonly used in applications which include monitoring of network traces, sensor data, stock quotes, web usage logs and call records [44, 129, 40]. Taking this approach, v_{11} and v_{12} will be considered to be synchronous if $|t_i\text{-}t_j| \leq \delta$, where t_i and t_j are the time at which v_{11} and v_{12} were observed, respectively, and δ is the length of the window. So, the exact-matching can be considered as a special case of window-based matching. In our work, we assume that the collected data is combined into a single instance as in Table 5.1, with their time differences resolved.

5.3 Missing Values Imputation Methods

Imputation is a term that refers to the process of estimating missing data of an observation based on available values of other variables [41]. It has become one of the most popular methods for solving missing value problems in survey data analyses. However, imputing missing values is not without danger. Dempster and Rubin [26] commented that data imputation is a general and flexible method for handling missing data problems, but is not without its pitfalls. They stated that caution is needed when employing imputation methods, otherwise the imputation will prove to be more problematic than leaving the data with missing values. This is because such methods may generate substantial biases between real and imputed data and consequently misleading results from the data.

Imputation methods work by substituting replacement values for the missing data, hence increasing the amount of usable data. Sande [121] discussed the problem of data imputation methods and concluded that they should satisfy three rules:

Rule 1. The method should not change the distribution of the dataset.

Rule 2. The method should retain the relationship among QoS attributes.

Rule 3. The method must not be too complex or computationally costly to apply.

Note that these three rules are desirable, but may not be simultaneously satisfied [139]. For example, to produce high quality imputation (e.g. retaining distribution and relationship) a method will usually be more computationally expensive. Hence, it is more important to consider application requirements when designing a missing value imputation method, rather than pursuing these rules in practice.

It is often difficult to determine and compare the accuracy of different imputation methods. This is because the same imputation method may give higher predictive accuracy rates in certain circumstances and not in others. Unless one is able to know the true

values of the missing data, it is difficult to determine the prediction accuracy of the imputation methods. In the literature, some studies of imputation methods have performed imputations on existing data sets with missing data [108], but because the true values were not known, the accuracy of the results could not be determined. Other studies have used real data sets and simulated missing data by deleting values, so the true value was known and the method's accuracy could be determined [36, 80].

For illustration purposes, we consider a simple example to compare the existing imputation methods w.r.t. the three rules given above. Suppose that we have a set of QoS data collected for service S that has two attributes A_1 and A_2, and the collected data is shown in Table 5.2, where asynchronous data is represented as missing values by $\perp$ and our aim is to estimate $\perp$. Different strategies may be followed [105, 39, 73, 98]. In this section, a number of missing values imputation methods are presented. The first five methods are considered simple because they are conceptually straightforward and require minimal computations. Other methods are considered to be advanced since they are conceptually more complex. Our intention is to use a suitable method that can produce a good estimation to handle asynchronous data. That is, an imputation method that would preserve the quality package's information (i.e. consider interaction among the multiple attributes of a service when some multiple levels of quality exist).

Table 5.2: The collected QoS data for S

time	A_1	A_2
1	0.3	0.4
2	0.7	0.8
3	0.3	0.4
4	0.3	0.4
5	0.7	$\perp$
6	0.7	0.8

5.3.1 Random Imputation

Random Imputation (RI) is an imputation method in which a missing value is replaced by a value randomly drawn from the set of available values [53]. In our example, if we assume that QoS data falls between 0 and 1, then we can randomly generate replacement values between 0 and 1, with all values having an equal probability of being generated. The RI method is simple and computationally efficient (**Rule 3**). However, it does not consider data distribution (**Rule 1**), nor any other relevant properties such as correlation between multiple attributes (**Rule 2**). Accordingly, imputing asynchronous data by random values may introduce bias and noise to the original data and give a misleading assessment result.

5.3.2 Most Common Value Imputation

Most Common Value (MCV) Imputation is one of the simplest methods to deal with missing values [106]. The value of the attribute that occurs most often is selected to be the value for all the missing values of that attribute. Principally, MCV imputation replaces the missing value with the mode. A problem with using the most common value as the replacement value is that the distribution of the dataset may have several modes. In this case, the missing values may be replaced by randomly selecting one of the modes [57]. If the mode corresponds to unobserved values with high frequency compared to other values, the percentage of correct imputations will be high. Otherwise, the imputation performance decreases and consequently the relative error of incorrectly imputed values will be high. In our example in Table 5.2, 0.4 is the most common value and so $\perp$ will be replaced by this value. Assuming that S delivers only two packages of quality w.r.t. A_1 and A_2, <0.3,0.4> and <0.7,0.8>, the imputed value makes row 5 inconsistent with the packages delivered by S and thus affects the relationship between attributes (**Rule 2**). Although the MCV method is computationally effective (**Rule 3**), replacing missing values by the most common value may change the distribution of the

dataset (**Rule 1**). If we allow imputed values to be used in predicting future missing values with this method, then over time the most common value could be the imputed value. This leads to a distortion of the distribution of the data and means the imputed value is less fit for use in calculating the assessment.

5.3.3 Last Observation Carried Forward Imputation

A method that has received considerable attention is the last observation carried forward (LOCF) imputation [93, 43, 118]. This approach is regularly used in epidemiological research, especially in clinical trials [50, 93]. The lack of explicit modelling assumptions has perhaps led to the popularity of this approach in the clinical trials arena where it is desirable to draw inferences based on as few assumptions as possible [23]. With the LOCF method, if an observation at a certain data collection point is missing, the last observed value is then used as an estimate for this missing observation. It works on the assumption that the outcome remains constant at the last observed value. Recent empirical studies have cautioned against the use of the LOCF technique [23, 131, 11] and have demonstrated its bias [118]. Such bias mainly stems from the unrealistic assumption that outcomes would not have changed from the last observed value, but in reality there is a change and larger differences between real values and imputed values may be artificially produced. Applying the LOCF method in our running example will suggest replacing $\perp$ by 0.4 (i.e. 0.4 in row 4 is the last observed value). Assuming that S delivers only two packages of quality w.r.t. A_1 and A_2, <0.3,0.4> and <0.7,0.8>, it is obvious that replacing $\perp$ by 0.4 is inaccurate since this value is expected to be delivered with 0.3 for attribute A_1, not 0.7 as row 5 suggests. In general, the LOCF method is computationally efficient (**Rule 3**). However, replacing the missing values by the most common value does not only change the distribution within the dataset (**Rule 1**), but also influences the relationship between multiple attributes (**Rule 2**).

5.3.4 Mean Imputation

Mean imputation is a widely used method for dealing with missing values [16, 8, 15, 51]. With this method, a missing value of an attribute is replaced by the mean of the values of that attribute. For example, in Table 5.2, $\perp$ will be replaced by 0.56, the mean of all values of A_2. This method is computationally efficient (**Rule 3**). However, according to Little and Rubin [78], it suffers from the following drawbacks:

1. The distribution of new values is an incorrect representation of the population values, because the shape of the distribution is distorted by adding values equal to the mean (**Rule 1**).

2. Observed correlations are depressed due to the repetition of a single constant value (**Rule 2**).

This method is particularly problematic when dealing with multiple attributes. In our example, the mean imputation method replaces $\perp$ by 0.56, if we assume S delivers only two distinct levels for attribute A_2 (0.4 or 0.8), then replacing $\perp$ by 0.56 is clearly inappropriate, as it is far away from the two distinct levels delivered by S. So, in this case, imputing missing values by the mean will actually be worse than leaving the data with missing values.

5.3.5 Median Imputation

Since the mean is affected by the presence of outliers, it seems natural to use the median [3, 57]. In the median imputation method, the missing values for a given attribute are replaced by the median of all known values of that attribute, so this method is computationally efficient (**Rule 3**). This method is a recommended choice when the distribution of the values of a given attribute is skewed [3]. In the case of a missing value in a categorical feature we can use mode imputation instead of either mean or median imputation. As with all previously single-value imputation methods, this method disturbs

the distribution of the data (**Rule 1**), since the same value is used to replace all missing values [37]. In our example in Table 5.2, $\perp$ in row 5 will be replaced by 0.4. Note that the correlation structure of the data in Table 5.2 is not being considered by this method (**Rule 2**). From the assessment point of view, replacing all missing values by 0.4 will mistakenly suggest that S can provide 0.4 for attribute $A2$ and 0.7 for attribute $A1$. Assuming S provides only two packages w.r.t. attributes A_1 and A_2, <0.3,0.4> and <0.7,0.8>, the imputed values by this method are clearly inaccurate and QoS assessment that uses these values will produce a misleading result.

5.3.6 Machine Learning Methods

With the advent of new computational methods, machine learning techniques have become increasingly attractive to researchers in the biomedical, behavioural, and social sciences, whose investigations are hindered by missing data [14, 56, 99, 20]. Imputation methods based on machine learning are sophisticated procedures that generally aim to learn from training examples to predict future events. In this approach, the observed values from other attributes are used as input to predict the missing value. An important argument in favour of these methods is that, frequently, attributes have relationships (correlations) among themselves and these correlations can be used to predict the missing values. In contrast to all previous methods, imputation methods based on machine learning can satisfy **Rule 2** by retaining the relationships among QoS attributes. Also, these methods do not change the distribution of the dataset (**Rule 2**). However, compared to the previously discussed methods, machine learning methods are more complex and computationally inefficient (**Rule 3**).

Broadly, two approaches may be followed to estimate missing values using machine learning techniques: instance- and model-based [107, 29]. Instance-based methods compare the instance containing a missing value to all the "complete" instances, and determine the missing value by measuring the similarity between them. These methods are desirable for their simplicity, good performance and robustness, but are computa-

tionally expensive and require substantial storage. Model-based methods, on the other hand, attempt to construct an explicit model from the instances and to use this model to determine the missing value. These methods, compared to the instance-based methods, are both computation and space efficient, but constructing an unbiased model is not straightforward.

In this work, we adopt the instance-based approach to estimating missing values. This is because first, these methods are well suited for estimating numerical value [137, 28, 30], which is the type of data we deal with. Second, in contrast to model-based methods, these methods offer a finer degree of approximation even when a small amount of data is used, so their estimate accuracy is often less erratic. Finally, these methods suffer less from over-generalisation than the model-based methods tend to do [29].

5.4 Handling Asynchronous Data using the kNN method

Several studies have found that the k nearest neighbour (kNN) method performs well or even better than other methods, both in the computing context [97, 132, 18, 140] and in other application areas such as health care and biology [138, 12, 8, 80, 72]. Motivated by the results reported by these studies, we employ a kNN algorithm to predict the missing values resulted from asynchronously monitored data. Since kNN automatically takes correlation within the data into account, it is good for handling scenarios where data are grouped, reflecting the existence of multiple quality packages. In the next section, we demonstrate how the missing values resulted from asynchronously monitored data may be imputed using a kNN algorithm.

5.4.1 kNN Algorithm

kNN is one of the simplest instance-based methods that makes a decision on a case based on the majority vote of its k-nearest neighbours [31]. To use it for handling

asynchronous data, we find the k synchronous instances that are closest to the instance containing a missing value, and then average the k synchronous instances as the estimated value for the missing one. Algorithm 5.1 shows how it works.

Algorithm 5.1: k-Nearest Neighbour

input: D_{syn}, a set of synchronous instances in D.

k, the number of considered neighbours.

$I_{asyn} = \langle a_1, a_2, \ldots, a_{j-1}, \perp, a_{j+1} \ldots, a_m \rangle$,

where $\perp$ represents a missing value.

output: v, the estimated value for $\perp$

1. $E \leftarrow \varnothing$
2. **for** $s = 1$ **to** $|D_{syn}|$ **do**
3. $\quad E \leftarrow E \cup DC(I_i \in D_{syn}, I_{asyn})$
4. $N_k \leftarrow \min_k(E)$
5. $v \leftarrow avg(N_k, A_j)$
6. **return** v

For an asynchronously collected service delivery instance that does not contain a value for attribute A_j:

$$I_{asyn} = \langle a_1, a_2, \ldots, a_{j-1}, \perp, a_{j+1} \ldots, a_m \rangle$$

we calculate its similarity with every instance in D_{syn}, the set of synchronous instances collected so far, using a distance function DC, and store the result in E (lines 2-3). From E, we select k shortest distances as the k nearest neighbours to I_{asyn} and store it in N_k (line 4). Finally, we estimate the missing value in I_{asyn} by averaging the k instances in N_k w.r.t. attribute A_j (line 5). To apply Algorithm 5.1 to estimate asynchronous data, some issues need to be considered. We discuss these issues in the following sections.

5.4.2 Measuring Distance

Loosely speaking, estimating missing values by a kNN method proceeds from the assumption that "similar problems have similar solutions". Thus, the performance of a kNN method depends critically on the similarity metric used. Various distance metrics have been proposed, e.g. Euclidean, Manhattan, Mahalanobis, Pearson, etc. Troyanskaya et al. [138] and Strike et al. [132] have demonstrated that for kNN Euclidean distance outperforms the others.

So, the implementation of DC in Algorithm 5.1 uses Euclidean distance:

$$DC(I_i, I_s) = \sqrt{\sum_{j=1}^{m}(v_{ij} - v_{sj})^2} \tag{5.1}$$

where v_{ij} and v_{sj} are the j-th values in I_i and I_s respectively, and m is the number of values that are present in I_i.

5.4.3 Neighbour Criteria

There are two strategies for selecting neighbours. The first strategy is in line with how the method is normally used, and allows only complete cases (i.e. the synchronous subset of the data) to be neighbours. This means that no incomplete cases (i.e. the asynchronous subset) can contribute to the substitution of a replacement value for a missing one. The second strategy allows all complete cases and certain incomplete cases to be neighbours. More specifically, a case can act as a neighbour if and only if it contains values for all attributes that the case being imputed has values for, and for the attribute being imputed. In our work, we use the first strategy. That is, we do not include asynchronous instances in searching for neighbours, nor use the estimated values to estimate other values. This is a conservative strategy which minimises errors in estimation.

5.4.4 Choosing the k value

An important issue for a kNN algorithm is how to choose an optimal value for k (i.e. number of considered neighbours). Different k can affect the performance of a kNN algorithm. Duda and Hart suggest the use of $k \approx \sqrt{N}$ [31], where N corresponds to the total number of all neighbours. Cartwright et al., on the other hand, suggest a low k, typically 1 or 2, but point out that $k = 1$ is sensitive to outliers and consequently use $k = 2$ [18]. Several others use $k = 1$, for example, Myrtveit et al. [97], Strike et al. [132], Huisman [53], and Chen and Shao [20]. Batista and Monard, on the other hand, report on $k = 10$ for large data sets [7]. As k increases, the mean distance to the target case (i.e. the case with missing value(s)) gets larger, which implies that the replacement values can be less precise. Revisiting the example introduced in Section 5.3, the estimation of $\perp$ will differ with different k. If we allow k to be 2, then the two data instances in rows 2 and 6 will be selected as the nearest neighbour for row 5 and averaged to give 0.8. However, if we allow k to be 5 (i.e. all data included), the missing value will be imputed by 0.56, which is the same value imputed by the mean imputation in Section 5.3.4.

Unfortunately, there is no theoretical criteria for selecting the best k, and it is usually determined empirically. In our case, a small k can deteriorate QoS assessment performance as it may over-emphasise on a few "close", but possibly not representative instances. On the other hand, a large k may include too many instances which may not be close enough to the row that is considered. When multiple quality packages exist, this may result in some unwanted overlapping between packages, resulting poor assessment accuracy. We will study this further in the next chapter.

5.4.5 Discussion

In this section, we highlight the advantages of using the kNN method to handle asynchronous data and study its limitations.

The main advantage of a kNN method is its simplicity. For this reason, researchers studying pattern recognition [49], DNA microarray data [138] and survey data [20] have used kNN to predict missing values. In addition, a kNN method does not make any assumptions about the underlying data distribution. This is useful as, in practice, data does not always obey the theoretical assumptions made about distributions. Moreover, there is no need to create a predictive model for each attribute with missing data [31]. In fact, a kNN method does not create explicit models (like a decision tree or a set of rules). Also, a kNN method can easily be used to predict examples with multiple missing values. Another main benefit of using a kNN method in our context is that it automatically takes correlation within the data into account, so it is good for handling scenarios where data are grouped, reflecting the existence of multiple quality packages.

One of the main drawback of a kNN method is that, whenever a kNN looks for the most similar instances, it searches through the entire dataset. This limitation can be critical for large databases. Many works that aim to address this limitation can be found in the literature. One of the most well-known solutions is to reduce the training data set, so instead of searching through the entire dataset, searching for neighbours will be limited to some prototypical examples [146]. The other way is to build a data structure. For example, in [19], a tree-like structure named RecTree was introduced to improve the efficiency of instance-based methods. A hierarchical clustering algorithm was performed to construct the tree. As a result, the search for neighbours was faster than scanning the entire dataset. In our work, however, we do not consider computational efficiency, as our main concern is to improve QoS assessment over multiple attributes w.r.t. accuracy and confidence. Notwithstanding, our proposed method is not affected by the computational efficiency of the kNN method. This is because the kNN method is only used to prepare and transform the asynchronous data into a synchronous form prior to, rather than during the QoS assessment.

5.5 SE+kNN

In this section, we describe our proposed method for QoS assessment which overcomes the limitations of existing methods. Instead of using only the synchronous subset of the collected instances as the SE method does (see Section 3.3.3), we propose to employ a kNN based technique to impute asynchronous data before using the data in assessment, and we call our method SE+kNN. Handling asynchronous data in such way can be considered as adding a data preparation stage to the QoS assessment process given in Chapter 3, as shown in Figure 5.1.

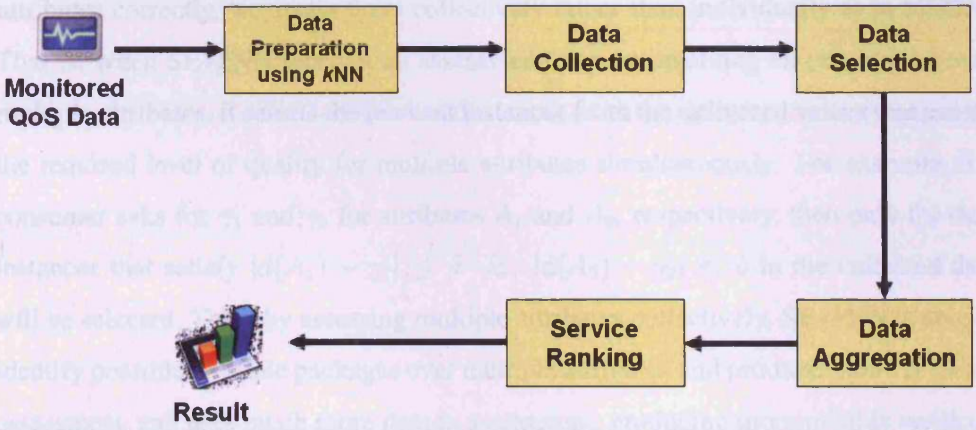


Figure 5.1: Integration of data preparation into QoS assessment process

The details of our data preparation process are given in Figure 5.2. For each new observed instance I, we check if I is synchronous or not. If it is synchronous, we create two copies of I, one is recorded in the Synchronous Data part which will be used by the kNN component given in Algorithm 5.1, and the other is stored in Data Collection to be used in QoS assessment. If I is not synchronous, it has at least one missing value, we pass it to the kNN algorithm to transform it into a synchronous form. When this is done, it is passed to Data Collection to be used in QoS assessment.

For the actual assessment, SE+kNN is an extension of MQSM. That is, SE+kNN goes

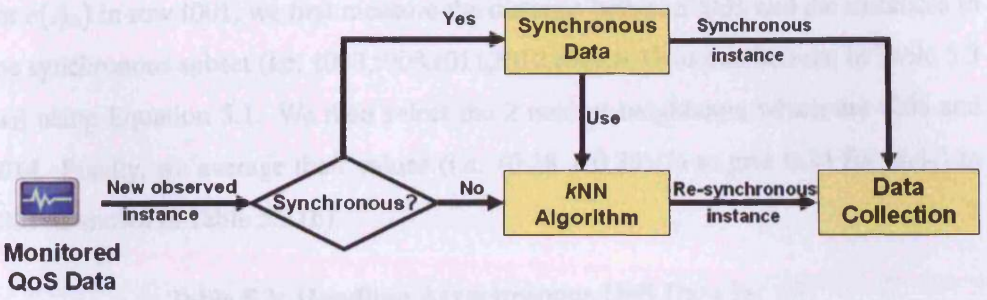


Figure 5.2: Data preparation process

through the four stages described in Section 3.3.2. However, to deal with multiple attributes correctly, we assess them collectively rather than individually as in MQSM. That is, when SE+kNN receives an assessment request involving an expectation over multiple attributes, it selects the relevant instances from the delivered values that satisfy the required level of quality for multiple attributes simultaneously. For example, if a consumer asks for γ_1 and γ_2 for attributes A_1 and A_2, respectively, then only the data instances that satisfy $|d(A_1) - \gamma_1| \leq \delta \;\; \& \;\; |d(A_2) - \gamma_2| \leq \delta$ in the collected data will be selected. Thus, by assessing multiple attributes collectively, SE+kNN is able to identify possible multiple packages over multiple attributes and produces more accurate assessment, and uses much more data in assessment, producing more reliable verdicts.

To show how the SE+kNN method works, we revisit the example introduced in Section 3.3 to assess $S1$ w.r.t. A_1 and A_2 based on a consumer's expectation $\gamma_1 = 0.4$ and $\gamma_2 = 0.4$. For better explanation, we have expanded Table 3.3 by adding 5 more instances (i.e. $t011$ - $t015$), as shown in Table 5.3 (a). First, our kNN algorithm is used to transform the asynchronous data in Table 5.3 (a) with k=2. This results in the replacements for the "missing" values as shown in Table 5.3 (b), where the *bold* instances represent the data collected from the monitoring tools and the other values represent the data predicted by the kNN algorithm. The missing expectation values in Table 5.3 (a) are predicted similarly: we estimate the missing $e(A_i)$ by selecting its k nearest neighbours and then taking their average. For example, to predict the expected value

for $e(A_2)$ in row $t001$, we first measure the distance between $t001$ and the instances in the synchronous subset (i.e. $t003$,$t008$,$t011$,$t012$,$t014$,$t015$ as can be seen in Table 5.3 (a)) using Equation 5.1. We then select the 2 nearest neighbours which are $t008$ and $t014$. Finally, we average their values (i.e. (0.28 + 0.39)/2) to give 0.34 for $e(A_2)$ in $t001$ as shown in Table 5.3 (b).

Table 5.3: Handling Asynchronous QoS Data for $S1$

(a)

TID	$\langle e(A_1), d(A_1)\rangle$	$\langle e(A_2), d(A_2)\rangle$
t001	**⟨0.27, 0.35⟩**	
t002		**⟨0.35, 0.41⟩**
t003	**⟨0.77, 0.67⟩**	**⟨0.85, 0.72⟩**
t004	**⟨0.41, 0.37⟩**	
t005		**⟨0.21, 0.38⟩**
t006	**⟨0.65, 0.71⟩**	
t007		**⟨0.80, 0.83⟩**
t008	**⟨0.36, 0.31⟩**	**⟨0.28, 0.47⟩**
t009	**⟨0.69, 0.65⟩**	
t010		**⟨0.87, 0.87⟩**
t011	**⟨0.43, 0.40⟩**	**⟨0.27, 0.35⟩**
t012	**⟨0.70, 0.72⟩**	**⟨0.79, 0.85⟩**
t013		**⟨0.77, 0.70⟩**
t014	**⟨0.33, 0.37⟩**	**⟨0.39, 0.43⟩**
t015	**⟨0.60, 0.64⟩**	**⟨0.75, 0.79⟩**

(b)

TID	$\langle e(A_1), d(A_1)\rangle$	$\langle e(A_2), d(A_2)\rangle$
t001	**⟨0.27, 0.35⟩**	⟨0.34, 0.45⟩
t002	⟨0.35, 0.34⟩	**⟨0.35, 0.41⟩**
t003	**⟨0.77, 0.67⟩**	**⟨0.85, 0.72⟩**
t004	**⟨0.41, 0.37⟩**	⟨0.28, 0.41⟩
t005	⟨0.4, 0.39⟩	**⟨0.21, 0.38⟩**
t006	**⟨0.65, 0.71⟩**	⟨0.77, 0.79⟩
t007	⟨0.74, 0.68⟩	**⟨0.80, 0.83⟩**
t008	**⟨0.36, 0.31⟩**	**⟨0.28, 0.47⟩**
t009	**⟨0.69, 0.65⟩**	⟨0.82, 0.76⟩
t010	⟨0.74, 0.68⟩	**⟨0.87, 0.87⟩**
t011	**⟨0.43, 0.40⟩**	**⟨0.27, 0.35⟩**
t012	**⟨0.70, 0.72⟩**	**⟨0.79, 0.85⟩**
t013	⟨0.65, 0.66⟩	**⟨0.77, 0.70⟩**
t014	**⟨0.33, 0.37⟩**	**⟨0.39, 0.43⟩**
t015	**⟨0.60, 0.64⟩**	**⟨0.75, 0.79⟩**

Second, to assess $S1$ with respect to $\gamma_1 = 0.4$ and $\gamma_2 = 0.4$, SE+kNN selects instances from Table 5.3 (b) based on $|d(A_1) - 0.4| \leq \delta \quad \& \quad |d(A_2) - 0.4| \leq \delta$ which gives us $\{t001, t002, t004, t005, t008, t011, t014\}$, assuming $\delta = 0.1$. For this set, we have $\alpha_{min} = 0.27$ and $\alpha_{max} = 0.43$ for attribute A_1, and $\alpha_{min} = 0.21$

and $\alpha_{max} = 0.39$ for attribute A_2. We then retrieve a new set of instances based on $0.27 \leq e(A_1) \leq 0.43$ & $0.21 \leq e(A_2) \leq 0.39$, which gives us the same set $\{t001, t002, t004, t005, t008, t011, t014\}$. Finally, we aggregate the delivered values in this set to find $QoS(A_1) = 0.36$ and $QoS(A_2) = 0.41$. Assuming the two attributes are equally important, the overall prediction for $S1$ is computed to be $QoS(S1) = 0.39$.

To calculate the confidence on a prediction made by SE+kNN, we need to calculate Rel_ω and Rel_θ. For illustration purposes, we only show how the confidence for the prediction of A_1 is calculated. Using Table 5.3 (b) and assuming that m_{min} is estimated to be 5, Rel_ω is calculated using Equation 4.3 which gives 1.0. To calculate Rel_θ, given the data selected by SE+kNN (i.e.$\{0.35, 0.34, 0.37, 0.39, 0.31, 0.40, 0.37\}$) and its prediction $p_1 = 0.36$ and assuming that $\epsilon_\theta = 0.05$, successful deliveries are determined by Equation 4.6. It is easy to verify that all delivered values are classified as successful because they are in the confidence range [0.31,0.41], determined by Equation 4.6. Now, we calculate α and β ($\alpha = 8$ and $\beta = 1$) and then using Equation 4.5 to calculate Rel_θ which gives 0.89. The overall confidence is computed using Equation 4.8 and the result is 0.89. So, by handling asynchronous data using the kNN algorithm, SE+kNN is able to produce higher confidence than SE does in QoS assessment (i.e. SE gives similar prediction, $QoS(A_1) = 0.36$, but with low confidence 0.45). This consequently helps consumers in selecting their preferred services. We will study this further in the next chapter.

5.6 Summary

Using data collected by monitoring tools in QoS assessment can eliminate several problems associated with user ratings and advertisements. However, the collected QoS data may not be well prepared for the purpose of QoS assessment. For example, when QoS data across multiple attributes is not synchronously collected, the method using synchronous data only could have substantially less data to use. This will affect the

confidence of assessment, particularly when we have a large number of attributes.

In this chapter, we proposed a method to transform asynchronous data into a synchronous form. That is, we handled asynchronous data by treating them as a dataset containing "missing" values and attempt to estimate such missing values. To do so, several imputation methods were investigated and analysed in order to determine the best method to use to predict the missing values. The imputation methods were compared based on three rules: the method should not change the distribution of the dataset, the method should retain the relationship among QoS attributes, and the method must not be too complex or computationally costly to apply. In our case, a k nearest neighbour (kNN) method is used to impute asynchronous data, since it automatically takes correlation within the data into account. This method is good for handling scenarios where data are grouped, reflecting the existence of multiple quality packages in our case. One drawback of the kNN method is that the method searches the entire dataset to find k nearest neighbours in order to estimate unpaired asynchronous values. This is expensive, particularly when the data size is large. In our work, however, computational efficiency is not particularly an issue since our main concern is to improve QoS assessment over multiple attributes in terms of accuracy and confidence, and the kNN method is actually used in the data preparation stage, not in the QoS assessment.

The goal of missing value imputation usually extends beyond simply making an accurate estimation. In our case, the performance of imputation needs to be tested in the context of QoS assessment, for example, whether it leads to better service ranking. This will be discussed further in the next Chapter.

Chapter 6

Evaluation and Results

In this Chapter, an empirical evaluation of the QoS assessment approach developed in this thesis is presented. A set of experiments were designed and conducted to compare our proposed approach against other QoS assessment approaches. In our evaluation, simulation was used because it allows various factors to be controlled.

This Chapter is structured as follows. It begins with an overview of the evaluation process, in terms of the criteria used for evaluation and the range of scenarios used. This is then followed by the description of a software simulation developed to allow fine-grained control over a range of scenario parameters. The software is then used to carry out a range of experiments, to evaluate the performance of approaches in different scenarios. Finally, a summary of the findings of the empirical evaluation is presented.

6.1 Evaluation Methodology

To demonstrate the contribution from the approach developed in this thesis, it was necessary to demonstrate its effectiveness in improving service quality experienced by consumers. The quality experienced by consumers is, however, directly dependent on how accurately a QoS assessment method evaluates the quality that can be expected from candidate services. This in turn is dependent upon the reliability of the information obtained from QoS assessment methods. There are two aspects that must be considered in evaluating the reliability of the verdicts from the QoS assessment methods. The

first is the criteria that will be used for evaluating the performance of QoS assessment methods. These metrics provide a set of objective measures with which the effectiveness of assessment methods may be compared. The second is a set of scenarios which aim to test the methods under a range of parameters and assumptions. This will help demonstrate the conditions under which these methods are effective and establish any limitations. The evaluation space explored in this thesis is illustrated in Figure 6.1.

The three dimensions in Figure 6.1 provide guidelines for our evaluation. The set of scenarios (x-axis) is used as input to test the performance (y-axis) of a given QoS assessment method (z-axis). The three dimensions are described in the following sections. First, we review the QoS assessment methods that we will use in our evaluation. Then, a set of criteria is described to evaluate the effectiveness of these methods, including justification for the use of such metrics in our study. Finally, a set of scenarios is introduced to demonstrate the strengths and weaknesses of the QoS assessment methods in various situations.

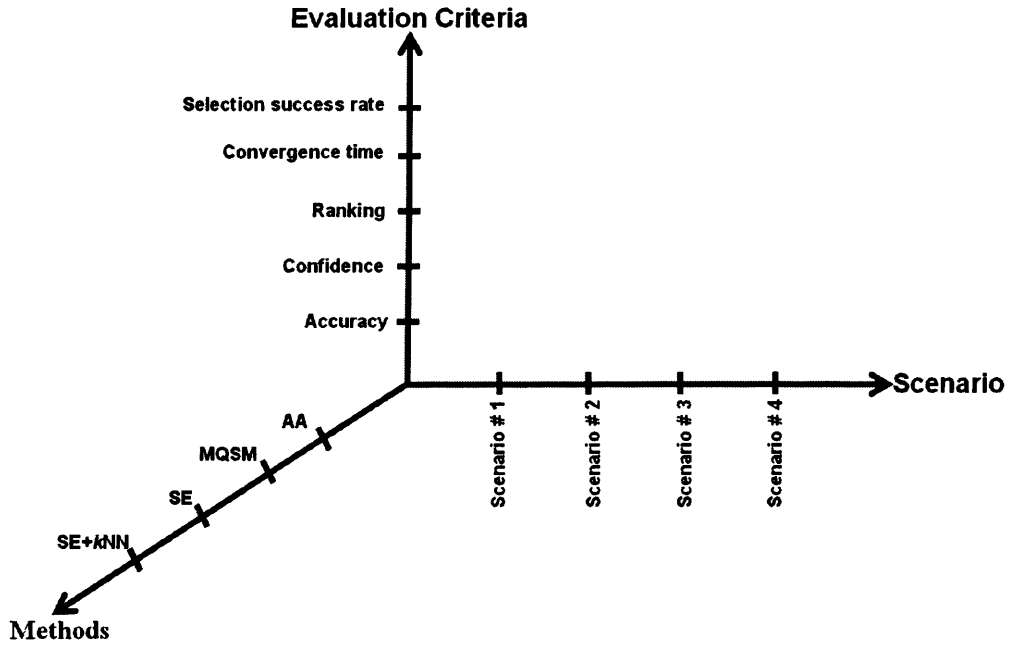


Figure 6.1: Evaluation Space

6.1.1 QoS Assessment Methods

In this section, we briefly re-cap the four QoS assessment methods shown in the z-axis in Figure 6.1. The objective of the experiments conducted in this chapter is to study the performance of the SE+kNN method proposed in this thesis against the Averaging All, MQSM ans SE assessment methods.

Averaging-All

This method averages all the data in each QoS attribute separately and then aggregates them into a single verdict [22, 81]. Conceptually, it does not apply any selection strategy as it simply assumes that all collected data are relevant for assessment. Section 3.3.1 has described the details of this method.

Multiple Quality Space Mapping (MQSM)

This method also processes each attribute separately, but attempts to identify possible quality packages within a single attribute [127]. In identifying any possible quality packages, this method goes through selection and mapping stages between delivered and expectation spaces as shown in Figure 3.3. It employs an expectation-based data selection strategy which has also been adopted by some other works in the literature [126, 149]. Section 3.3.2 has described the details of this method.

Synchronous Only Extension (SE) to MQSM

This method applies a simple "adjustment" to the MQSM approach discussed above. Instead of selecting and aggregating multiple attributes separately, this method assesses them collectively. In doing so, this method is capable of identifying the quality of a service that is offered as a package across multiple attributes (e.g. high `availability`

and quick `response time` offered together), but uses the synchronous subset of the data only. Section 3.3.3 has described the details of this method.

SE+kNN

This is the method that we have developed in this thesis. Instead of using only the synchronous subset of the collected QoS data as the SE method does, we treat the asynchronous collected data as containing missing values and employ a kNN based technique to impute the missing data before using it in assessment. So, effectively, we "add" kNN to the SE method to transform a set of asynchronous data into synchronous form first, and then use the SE method to calculate a QoS value. Section 5.5 has described the details of this method.

6.1.2 Evaluation Criteria

Before detailing our analysis results, we present several performance metrics which we use to evaluate and compare different QoS assessment methods. They are shown in the y-axis in Figure 6.1.

Assessment Accuracy

The accuracy of a QoS assessment method can be evaluated by observing how far away the predicted values are from the actual delivered levels for each QoS attribute. This metric is useful, because it gives an indication about the level of quality (conformance) a consumer may perceive from using a given service. By obtaining more accurate prediction from an assessment method, a consumer will be able to make a better decision about which service to choose.

Assessment Confidence

When a QoS verdict is derived for a given service using its past performance data, it is essential that we are able to establish its reliability. The measure of reliability is used to deal with uncertainty surrounding the likely behaviour of a service. This measure is useful because it helps to determine whether a service provider is more or less likely to deliver what a consumer expects. So a good QoS assessment should give an indication of the reliability of a QoS prediction. In our work, we provide a confidence value to reflect the quality of a given assessment method in predicting the future behaviour for a service. As we have described in Chapter 4, we calculate the reliability or confidence for a single attribute using both: data size (Rel_{ω}) and data deviation (Rel_{θ}). To compute a confidence for a service, we average the confidence for each attribute.

Service Ranking

If the goal of QoS assessment is to help a consumer to choose a preferred service among those functionally identical but quality-wise varying, then it is essential that we are able to rank a set of services at the end of assessment. So we also attempt to rank candidate services based on assessment outcome, using a combination of prediction and confidence. Improving assessment's accuracy and confidence will lead to better ranking and consequently better consumers' decision-making about which service to choose. Note that our ranking score is personalised in the sense that it is affected by the consumer's expectations.

Convergence time

This measure determines how quickly the assessment methods can converge to provide the information needed which enables selection of the appropriate service among the alternatives [90, 76]. This is the one that can best meet the consumer's requirements, i.e. it can deliver the closest level of quality to the requester's requirements and can do so

consistently. This measure is an important in a dynamic environment where services' behaviour change over time. The method that takes long time to converge may not be efficient in such an environment. So a good QoS assessment method should quickly converge to select the "best" candidate service for the consumer and then keep selecting it afterwards.

Selection success rate

This metric is used to measure the ratio of selecting the "correct" service by a given QoS assessment method. The correct service for each assessment requester is the one that not only delivers the closest level of service to the requester's expectation, but is also consistent in delivering that level over time (i.e. does not fluctuate a lot). Since we have configured services' behaviour in our simulation, it is easy to determine which service is the best for each assessment request. The rate of successful selections is calculated by dividing the number of correctly selected services by the total number of assessment requests. This metric is useful to evaluate the robustness of a QoS assessment method in dealing with varying behaviours of service providers (i.e. multiple levels of a service) and consumers (i.e. different expectations). It is an important measure, especially in a dynamic environment, where different requesters have different quality requirements and a service that is suitable for one consumer may not be suitable for another. A good QoS assessment method should therefore recognise the best candidate service for a specific assessment request.

6.1.3 Evaluation Scenarios

To support the argument that the proposed SE+kNN method leads to an improvement in QoS perceived by consumers, it is necessary to evaluate its performance w.r.t. the metrics given in Section 6.1.2 against a range of scenarios. These scenarios enable us to quantitatively compare the proposed method to other QoS assessment methods.

We designed four scenarios in total, as shown in Figure 6.1 by the x-axis, which were used to test the limitations of existing methods, especially in dealing with multiple attributes. Since it is difficult to cover all real world scenarios, we have selected some representative cases to test and demonstrate our main argument. These scenarios are derived from consumers' and providers' behaviours which are discussed below.

Service Consumer's Behaviour

Consumers' behaviours can be specified in terms of their QoS expectations. To make an objective assessment of a provider, a set of consumer requirements must be identified. These requirements provide important information for the QoS assessment method to determine a service level that the consumer wishes to receive. The consumer's behaviour can be considered to have expectation over either single or multiple attributes. It is also possible to differentiate between the behaviours of consumers in terms of the required level of service. That is, for a given QoS attribute, the requirement may vary from one consumer to another. For example, high `availability` and fast `response time` are important requirements for someone using a stock trading service. In general, we allow the required level of service to be high (e.g. 16 MB broadband service), moderate (e.g. 8 MB broadband service), or low (e.g. 2 MB broadband service) in our simulation. Table 6.1 summarises the parameters used in our simulation to configure consumers' behaviours where $e(A_i)$ represents the expectation of the consumer w.r.t. attribute A_i, with 0 representing the minimum level of expectation and 1 the maximum.

Table 6.1: The parameters used to configure consumers' behaviours

Dimension	Possible Values
Attribute	Single / Multiple
Level	$e(A_i) \in [0,1]$

Service Provider's Behaviour

Service providers' behaviours can be specified in terms of the level of service they provide to their consumers. Due to the varying QoS requirements from different consumers, service providers may provide different levels of the same service to different consumers. We therefore allow service providers' behaviours to either offer a single package of a service to all consumers, regardless of their expectations, or offer multiple packages of a service based on consumers' expectations. For a quality package, a specified level of service will be targeted by the provider. It is unrealistic to assume that the provider will always deliver this level. Therefore, we allow delivered values from each service package to have a normal distribution. In our simulation, we define each quality package according to a given mean (μ) and standard deviation (σ) to configure the provider's behaviour. The former determines the target level whereas the latter determines how good the provider has delivered this level. The value of μ may be low, moderate or high. A small value of σ indicates that the provider has consistently delivered a level of service close to μ. A large σ, on the other hand, means that the provider has fluctuated a lot in delivering μ to the consumers. Additionally, for each package, we define two further parameters to determine to which consumers the package will be offered based on their expectations. This gives us more control in the simulation. Table 6.2 summarises the parameters used in our simulation to configure an individual provider's behaviour, with 0 representing the minimum level of service and 1 the maximum.

Scenarios

The combinations of specific consumers' and providers' behaviours form a set of scenarios. These scenarios are used to test a QoS assessment w.r.t. a given metric. In our experiments, we considered the following four scenarios:

- **Scenario 1:** The service consumer has an expectation on only a single QoS at-

Table 6.2: The parameters used to configure providers' behaviours

Dimension	Possible Values
Package	Single / Multiple
Expectation	e(A_i) $\in$ [0,1]
Target Level (μ)	$\mu \in$ [0,1]
Consistency (σ)	$\sigma \in$ [0,0.5]

tribute and the service provider offers only a single package of service to all consumers, regardless of their expectations.

- **Scenario 2:** The service consumer has an expectation on only a single QoS attribute and the service provider offers multiple packages of service based on the consumer's expectations.

- **Scenario 3:** The service consumer has an expectation on multiple QoS attributes and the service provider offers only a single package of service to all consumers, regardless of their expectations.

- **Scenario 4:** The service consumer has an expectation on multiple QoS attributes and the service provider offers multiple packages of service based on the consumer's expectation.

These scenarios determine, at a high level, the behaviour of individual consumers and providers. However, for each scenario, the required level of service by the consumer may vary from low, to moderate to high. Similarly, the delivered level of service that is targeted by each provider may be different, but is based on consumers' expectations (see Table 6.2). Also, the provider's behaviour in delivering the targeted level may vary from being fluctuate (a large σ) to consistent (a small σ).

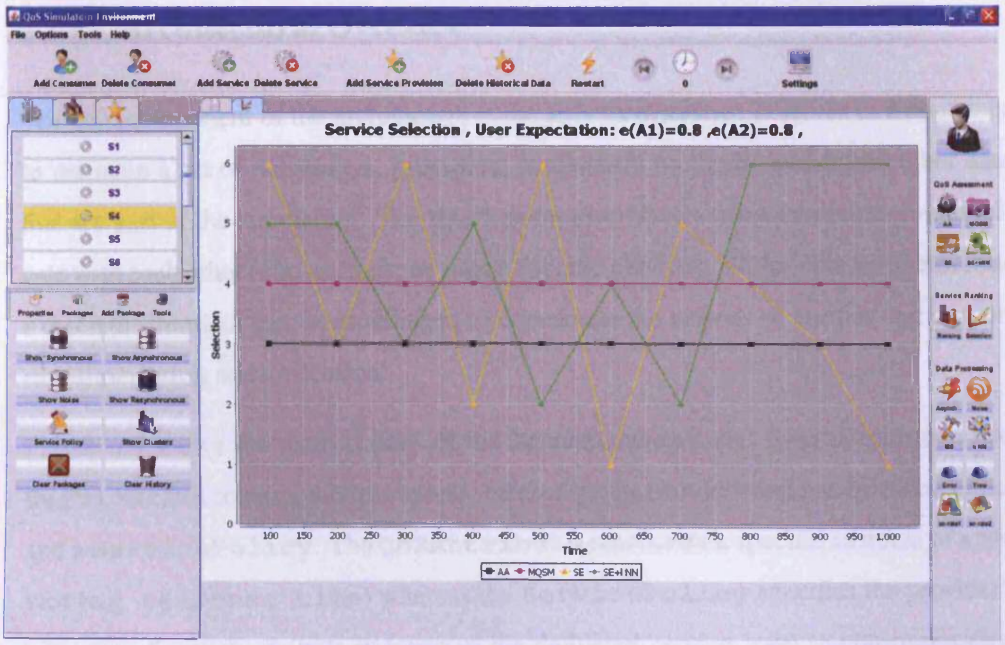


Figure 6.2: A snapshot of the GUI interface of the simulation interface

6.2 Simulation Environment

Evaluation of the effectiveness of QoS assessment methods within the scenarios described in Section 6.1.3 and using the performance metrics described in Section 6.1.2 requires fine-grained control over many aspects of a service provisioning environment. That is, we must be able to model consumers' behaviours in terms of their QoS expectations of quality, providers' behaviours in terms of service levels delivered to consumers, and service provision in which expectation and delivered values can be captured and recorded. Therefore, we implemented a simulation environment using the NetBeans IDE 6.7 [2] to simulate interactions between service consumers and providers. Using this tool, different scenarios can be easily constructed and simulated. It is also possible to obtain different results for different settings. Figure 6.2 provides a snapshot of the simulation interface.

6.2.1 Architectural Overview

The core component of the architecture is the **Environment**. It serves as a container to maintain a set of consumers, providers, assessment methods, events and other data that are part of the simulation. The **Environment** allows these entities to communicate with each other (e.g. a consumer uses a specific service). Within this **Environment**, a discrete simulator can be established to coordinate the actions of entities and support experimentation and evaluation.

Figure 6.3 shows the main classes of the **Environment** and their interaction. The **Service** class corresponds to a service offered by the provider and has **QoSAttribute** and a **ServicePolicy**. The **QoSAttribute** represents a specific attribute of a service (e.g. `response time`) whereas the **ServicePolicy** specifies the provider's behaviour for each attribute in terms of the delivered service level to consumers (e.g. single level of service at 0.5). The creation of a new service in our simulation corresponds to a new service being published. That is, the service is made public by registering itself in the **Environment** so it can be discovered by interested consumers.

The **Consumer** class is an entity in the **Environment** that sends an assessment request to the **QoSAssessment** class to specify the level of service required of the candidate services w.r.t. each **QoSAttribute**. The consumer's request is sent as a message to the QoS assessment method, and the message contains the expectation for each attribute. The QoS assessment method retrieves the relevant historical data for the candidate services and executes the assessment. The assessment result then ranks the services and the highest ranked service is recommended to the consumer.

If a request for a service is accepted by the provider, a service level agreement (SLA) which defines a set of consumer expectations is established and the consumer is given permission to start consuming the service. During service provision, the service provider will deliver a service level based on a policies predefined in the **ServicePolicy**. That is, a certain value (e.g. 0.37) will be generated by the **Monitor** class to simu-

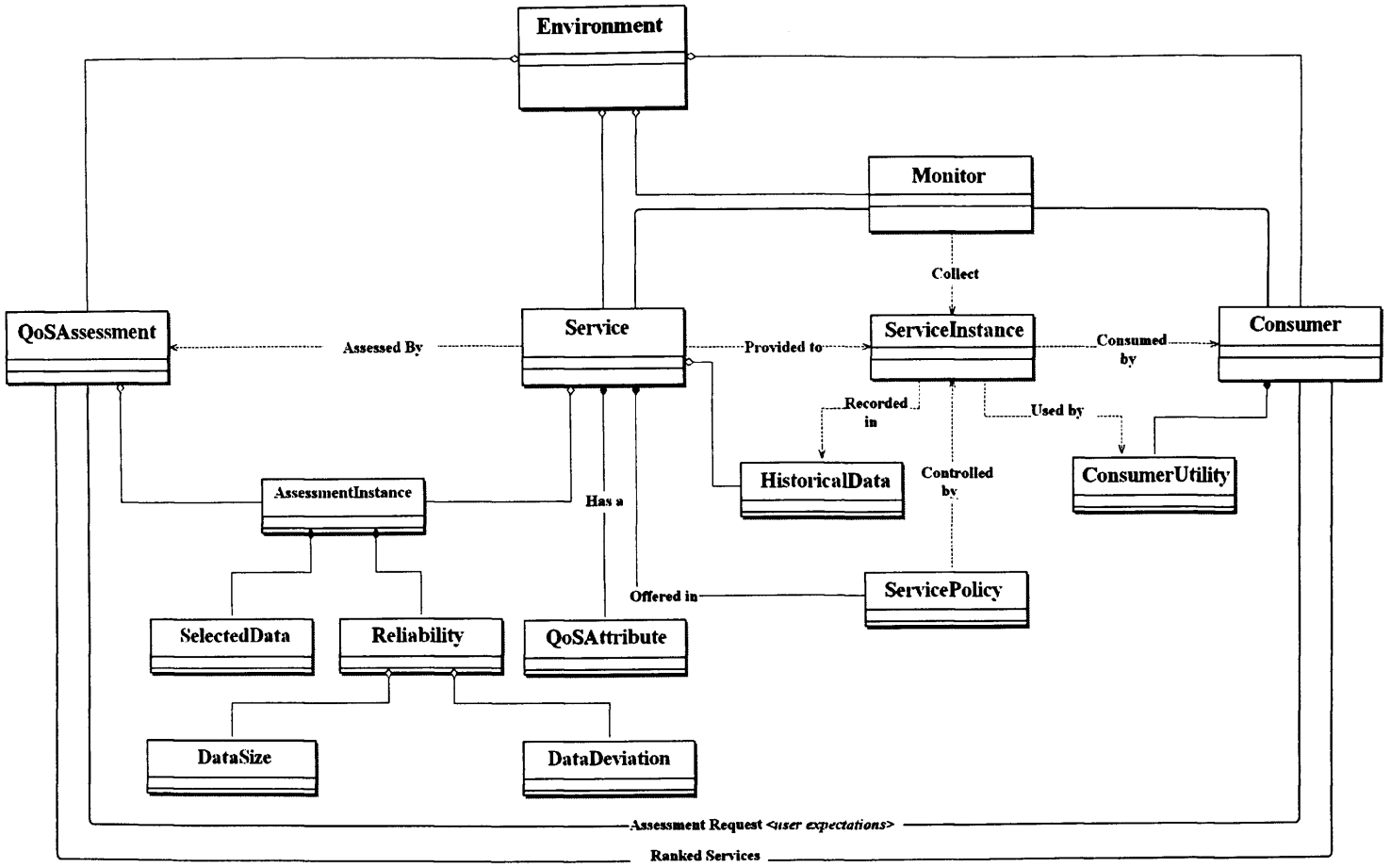


Figure 6.3: Main classes of the `Environment` simulation

late the level of service delivered to the consumer by the provider. This value, however, will be generated based on consumer's expectation which has been defined in the `ServicePolicy` during the configuration of the service's behaviour. When a service is provided to the consumer, the data about the instance is collected by the `Monitor` class. The information collected for each single service instance includes *time*, *expectation* and *delivered* level of service. The *time* is the time that the service provision instance was monitored, the *expectation* is the quality requirement of the service consumer at the *time*, and the *delivered* is the value observed by the `Monitor` at the *time*. The `ConsumerUtility` class then uses this information to calculate conformance, i.e. how well a particular provider met consumer's expectations. The `HistoricalData` class records this information for use by the `QoSAssessment` methods in the future to determine the level of service that might be delivered by the providers. It is important to emphasis that, in our simulation, the `QoSAssessment` has access to the historical data recorded in the `HistoricalData` of each service, but not to the `ServicePolicy`.

6.2.2 Control Component

A set of control components are provided in order to manage the properties of the entities described above and other aspects of the simulation, including:

1. **Simulation Properties:** Controls to manage the set of properties relating to the simulation environment. The behavioural aspects relating to the simulation could be defined, including the length of the simulation and the speed of simulation.

2. **Consumer Properties:** Controls to manage a set of services in the simulation environment. The consumers could be created and the aspects related to their behaviours could be specified. For example, we could add new consumers to the environment and determine their expectations on the level of a service.

3. **Service Properties:** Controls to manage a set of service providers in the simulation environment. The services could be added and their behaviours could be established. For example, we could create a new service provider and determine how many service levels will be provided to the consumers by the service.

4. **Assessment Method Properties:** Controls to manage the set of assessment methods described in Section 6.1.1 and set up the various parameters required by each method.

5. **Monitoring and Evaluation Properties:** The results of each assessment method in the environment can be monitored and viewed through a standard interface which provides information relating to predictions, confidence values, and ranking scores.

These controls can be used at the start of a simulation, or adjusted at any point during a simulation.

6.3 Experimental Results

To validate our assertions and verify the effectiveness of our approach, we conducted a set of experiments which tested and compared our proposed method against other methods using the scenarios described in Section 6.1.3. The experiments were designed to simulate interactions between service consumers and providers, and were conducted in the simulation environment described above where the actual behaviour of service providers could be accurately controlled and captured. The simulation allowed us to model providers' behaviours in terms of service levels delivered to consumers over time, and service provisions in which expectation and delivered values were recorded. The experimental setup is described below and then the results are presented and analysed.

6.3.1 Experimental Setup

A population of service consumers and providers were created and added to the environment. The behaviours of consumers and providers were configured to cover the scenarios outlined in Section 6.1.3. We assumed that all services provided the same functionality, and all consumers had the same functional requirements which were satisfied by these services. The QoS assessment was designed to work in this situation to help the consumers select the services that best meet their expectations. In other words, the QoS assessment methods are used to help in determining how likely it is that a given service would meet a consumer's expectation. To compare the performance of different QoS assessment methods, the experiments were conducted as if the consumer consulted each method independently.

Service Providers

Six service providers ($S1 - S6$) were used in our experiments. Each service they provide consisted of two attributes (A_1 and A_2) and offered different quality packages for different consumer expectations. The behaviour of these services was configured as in Table 6.3. That is, each time a service received a request from a consumer, it responded by generating a value that represented a delivered value to the consumer that satisfied the conditions given in Table 6.3. This value was normally distributed within each quality package according to a given mean (μ) and standard deviation (σ), as shown in Table 6.3.

Service Consumers

Ten consumers ($C1 - C10$) were used in our experiments. Each had different requirements in terms of either single or multiple attributes, and the level of service required by that attribute. These requirements are summarised in Table 6.4. As Table 6.4 shows, $C1 - C4$ were "single attribute" consumers who care only about the quality of service

Table 6.3: Service provider's behaviours configuration

Service	Package	$e(A_1)$	$d(A_1)$	$e(A_2)$	$d(A_2)$
$S1$	$P1$	[0.00, 1.00]	*Random*	[0.00, 1.00]	*Random*
$S2$	$P1$	[0.00, 1.00]	$N(0.50, 0.20)$	[0.00, 1.00]	$N(0.50, 0.20)$
$S3$	$P1$	[0.00, 1.00]	$N(0.50, 0.05)$	[0.00, 1.00]	$N(0.50, 0.05)$
$S4$	$P1$	[0.00, 0.50]	$N(0.20, 0.05)$	(0.50, 1.00]	$N(0.80, 0.05)$
	$P2$	(0.50, 1.00]	$N(0.80, 0.05)$	[0.00, 0.50]	$N(0.20, 0.05)$
$S5$	$P1$	[0.00, 0.50]	$N(0.30, 0.05)$	[0.00, 0.50]	$N(0.30, 0.05)$
	$P2$	(0.50, 0.80]	$N(0.75, 0.05)$	(0.50, 0.80]	$N(0.75, 0.05)$
	$P3$	(0.80, 1.00]	$N(0.95, 0.05)$	(0.80, 1.00]	$N(0.95, 0.05)$
$S6$	$P1$	[0.00, 0.30]	$N(0.10, 0.05)$	[0.00, 0.30]	$N(0.10, 0.05)$
	$P2$	(0.30, 0.50]	$N(0.40, 0.05)$	(0.30, 0.50]	$N(0.40, 0.05)$
	$P3$	(0.50, 0.70]	$N(0.60, 0.05)$	(0.50, 0.70]	$N(0.60, 0.05)$
	$P4$	(0.70, 1.00]	$N(0.90, 0.05)$	(0.70, 1.00]	$N(0.90, 0.05)$

w.r.t. that attribute. The consumers $C5 - C10$, on the other hand, were "multiple attributes" consumers because they had expectations on attributes A_1 and A_2. For each consumer request, we have shown in Table 6.4 the best service according to their behaviour in Table 6.3. For example, the best service for $C1$ is $S3$ because it delivers the required service level consistently.

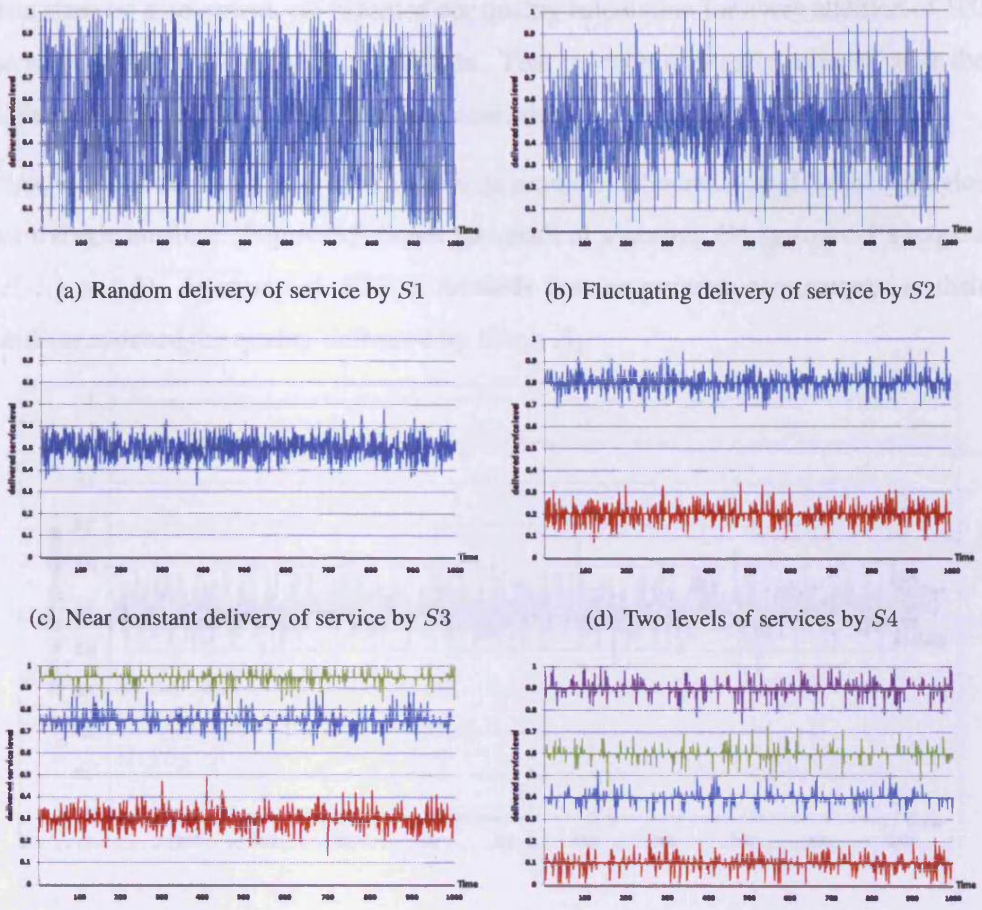
Data Generation

To test the performance of the four QoS assessment methods described in Section 6.1.1, for each of our experiments, we generated 1000 tuples in the form of $\langle e(A_1), d(A_1), e(A_2), d(A_2)\rangle$, each representing a past service instance. For example, $\langle 0.4, 0.5, 0.6, 0.7\rangle$ represents an observed service provision in which the expectations from the consumer on A_1 and A_2

Table 6.4: Service requester's expectations

Consumer	$e(A_1)$	$e(A_2)$	Best Service
$C1$	0.50		S3
$C2$		0.10	S6
$C3$		0.20	S4
$C4$	0.9		S6
$C5$	0.80	0.20	S4
$C6$	0.75	0.75	S5
$C7$	0.30	0.30	S5
$C8$	0.80	0.80	S5
$C9$	0.50	0.50	S3
$C10$	0.95	0.85	S6

of the service are 0.4 and 0.6, and the corresponding delivered qualities are 0.5 and 0.7, respectively. The values of these instances were generated to satisfy service behaviours given in Table 6.3. For example, for $S4$, we generated two packages, each consisting of 500 instances, and for its package $P1$, we generated expectations in $[0.0, 0.5]$ randomly, and their corresponding delivered qualities as a normal distribution in $[0, 1]$ with a mean $\mu = 0.2$ and a standard deviation $\sigma = 0.05$. Some example test data are shown in Figure 6.4. We allowed some variations in expectations in order to mirror the real world situations where consumers may broadly have similar expectations for a service quality, but vary slightly within that broad expectation. We also allowed the delivered values to have some "patterns" so as to simulate some real world scenario. For example, the set of data for $S3$ shown in Figure 6.4(c) exhibits a near constant service level, indicating a fairly stable delivery of service, whereas the set of data for $S2$ in Figure 6.4(b) exhibits a fluctuating delivery of service.



(a) Random delivery of service by $S1$

(b) Fluctuating delivery of service by $S2$

(c) Near constant delivery of service by $S3$

(d) Two levels of services by $S4$

(e) Three levels of services by $S5$

(f) Four levels of services by $S6$

Figure 6.4: Delivered qualities on A_1 by the six services

6.3.2 Accuracy of Assessment

In this section, we study prediction accuracy of the four QoS assessment methods (AA, MQSM, SE and SE+kNN) in different scenarios. The prediction accuracy can be evaluated by observing how far the predicted values are from the actual delivered levels for each QoS attribute. That is, we compare the prediction value produced by each QoS assessment method to the actual value delivered by the assessed service. We have carried out a set of simulations and report on the results. To observe the effect of different

data sizes on assessment, we repeated our quality calculation for every addition of 100 service instances using the four methods. That is, each service is assessed after the collection of 100,200, ..., 1000 service instances.

First, we considered the case where a service provider delivered a single level of service for a single attribute. Figure 6.5 shows the result of assessing $S3$ against $C1$'s request ($e(A_1) = 0.5$). As expected, all four methods gave an accurate assessment, i.e. their verdicts matched the quality delivered by $S3$ on A_1.

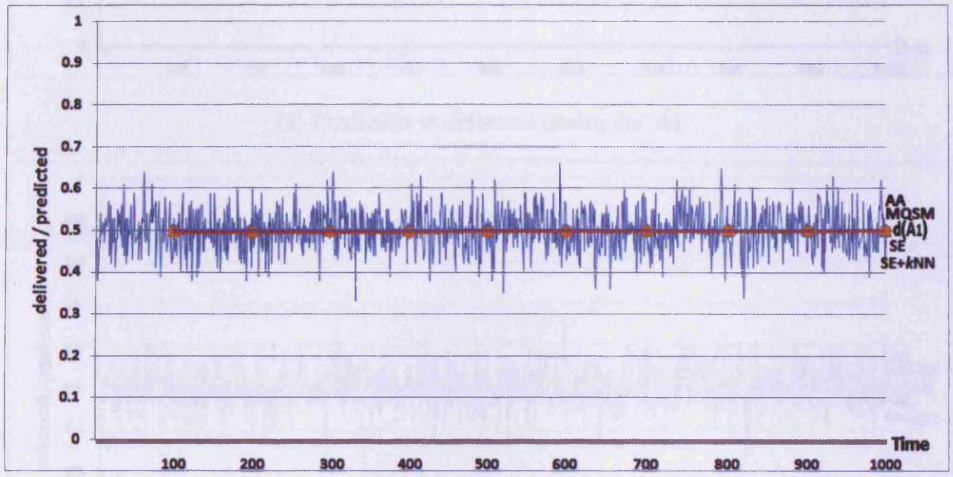
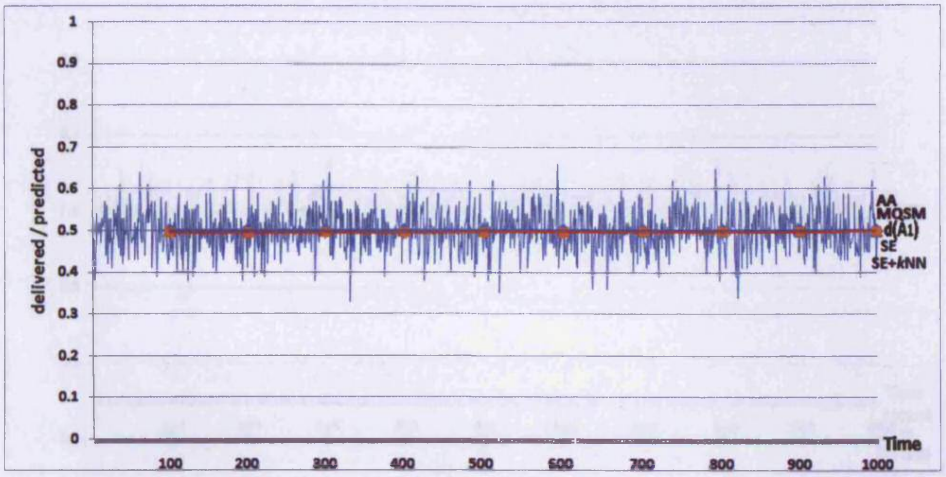
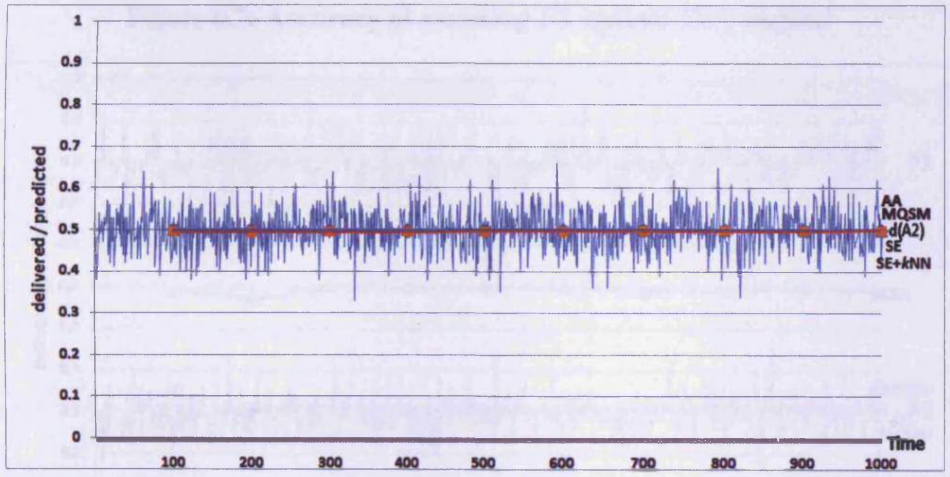


Figure 6.5: Accuracy of assessing $S3$ against $C1$'s request

We then considered services offering a single package over multiple attributes. In this experiment, we tested the assessment of $S3$ against $C9$'s request ($e(A_1) = 0.5, e(A_2) = 0.5$)) by the four methods, and the result is shown in Figure 6.6. Again, all methods performed well. Note that although AA gave a correct result in this case (i.e. it correctly predicted the quality that would be delivered by $S3$ on A_1 and A_2), it can easily give inappropriate values in other cases. For example, when assessing $S3$ against $C3$'s request, as shown in Figure 6.7, AA gave the same verdict as in Figure 6.5, since it does not take consumer expectations into account. The expectation-based methods (i.e. MQSM, SE and SE+kNN), on the other hand, assessed the quality of $S3$ dynamically by considering the level of service requested by $C3$ (i.e. $e(A_2) = 0.2$), and returned a



(a) Prediction vs delivered quality for $A1$



(b) Prediction vs delivered quality for $A2$

Figure 6.6: Accuracy of assessing $S3$ against $C9$'s request

default value of -0.1, meaning that no verdict could be reached. This is correct as $S3$ does not provide the service at the requested quality level according to Figure 6.4(c).

We also examined the assessment accuracy for services offering multiple levels of quality over single attributes. For this experiment, we set the four methods to assess $S4$ against $C3$'s request ($e(A_2) = 0.2$). The result is shown in Figure 6.8.

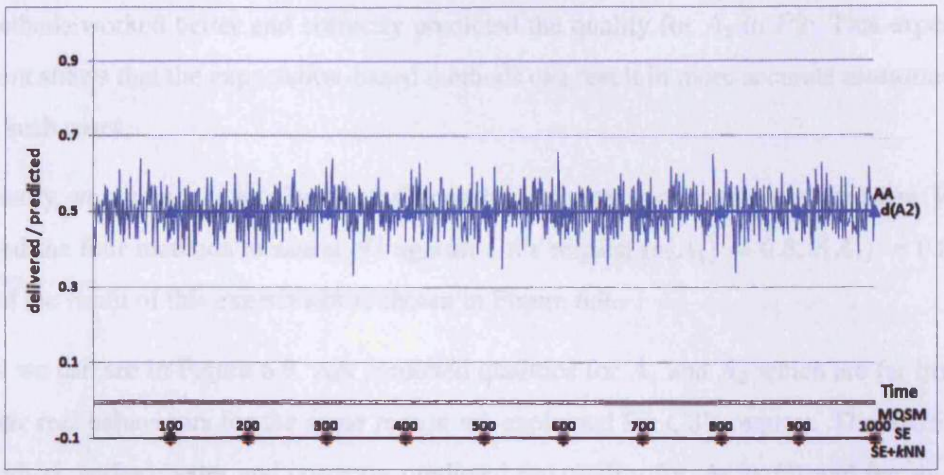


Figure 6.7: Accuracy of assessing $S3$ against $C3$'s request

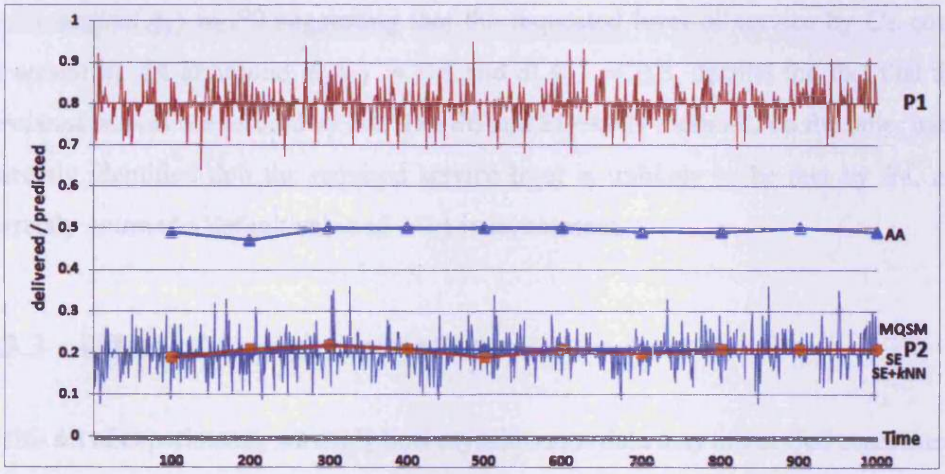


Figure 6.8: Accuracy of assessing $S4$ against $C3$'s request

As can be seen from Figure 6.8, AA predicted the quality for A_2 that is between the two packages ($P1$ and $P2$), which is not obtainable from $S4$ in practice. This was caused by the fact that there existed two levels of qualities for A_2 and each level was only available to consumer requests in certain ranges (see Table 6.3), but the method inappropriately aggregated them together in the assessment. The MQSM, SE and SE+kNN

methods worked better and correctly predicted the quality for A_2 in $P2$. This experiment shows that the expectation-based methods can result in more accurate assessment in such cases.

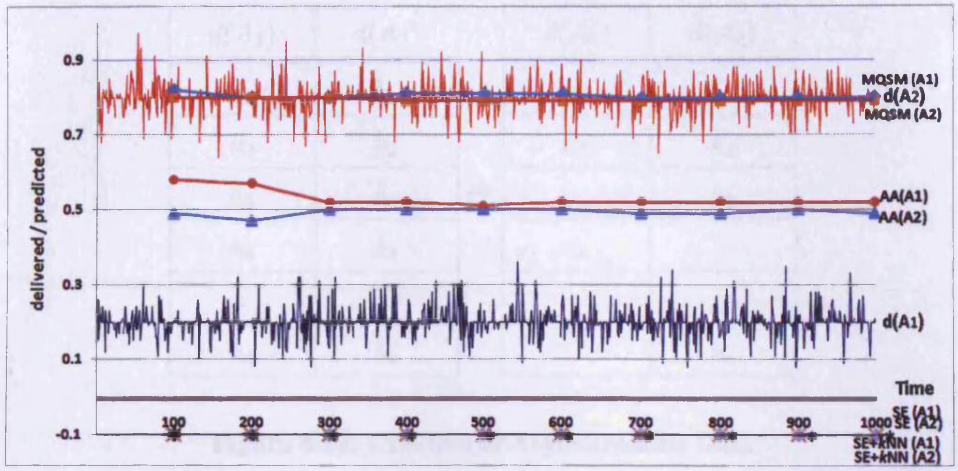
Finally, we studied the services that offer multiple packages over multiple attributes. We used the four methods to assess $S4$ against $C8$'s request $(e(A_1) = 0.8, e(A_2) = 0.8)$, and the result of this experiment is shown in Figure 6.9.

As we can see in Figure 6.9, AA predicted qualities for A_1 and A_2 which are far from their real behaviours for the same reason we explained for $C3$'s request. The MQSM method worked better and correctly predicted the quality for A_2 in $P1$ and for A_1 in $P2$, but failed to do so for the other attribute in the respective package correctly. This is because MQSM assessed two attributes individually, and then mistakenly paired $d(A_2)$ in $P1$ with $d(A_1)$ in $P2$ suggesting that the requested level of service by $C8$ could be served by $S4$ at around $d(A_1) = 0.8$ and $d(A_2) = 0.8$, despite the fact that this *combination* was not offered by $S4$. The SE and SE+kNN methods, on the other hand, correctly identified that the required service level is unlikely to be met by $S4$, and correctly returned a default value of -0.1 in assessment.

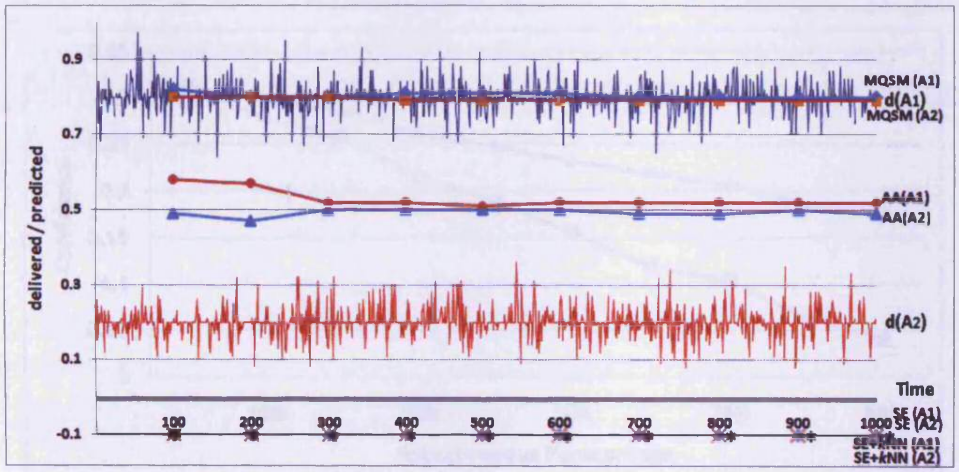
6.3.3 Effect of Asynchronous Data

In this set of experiments, we study how asynchronous data may affect QoS assessment. We compare SE and MQSM only in this study because AA is similar to MQSM as far as handling asynchronous data is concerned, and SE+kNN is not affected by the asynchronous data. We used the same dataset generated for the previous experiments, but randomly removed a certain percentage of values from the two attributes to simulate asynchronously collected data. That is, we randomly removed a percentage of $d(A_1)$ or $d(A_2)$ (but not both) from $\langle e(A_1), d(A_1), e(A_2), d(A_2)\rangle$ as illustrated in Figure 6.10.

Again, we considered the services that offer multiple packages over multiple attributes. This time, however, we assumed that $S4$ was to be assessed against $C5$'s request



(a) Prediction for Package $P1$



(b) Prediction for Package $P2$

Figure 6.9: Accuracy of assessing $S4$ against $C8$'s request

($e(A_1) = 0.8, e(A_2) = 0.2$), so that SE would return a non-default verdict and we could examine its confidence score. We set the required confidence level to be $\lambda = 0.99$ and maximum tolerable error level to be $\epsilon = 0.05$. In Figure 6.11, we plotted the confidence of the assessment made by SE and MQSM, where we averaged the confidence of two attributes into one, and the percentage of asynchronous data was varied from 10% to 90% with an equal amount of data removed from A_1 and A_2. The MQSM and SE

$d(A_1)$	$d(A_2)$
a_1	b_1
a_2	b_2
a_3	b_3
a_4	b_4
:	:
a_n	b_n

$\Rightarrow$

$d(A_1)$	$d(A_2)$
a_1	
a_2	b_2
	b_3
a_4	
:	:
	b_n

Figure 6.10: Creation of Asynchronous Data

correctly identified the package in this experiment.

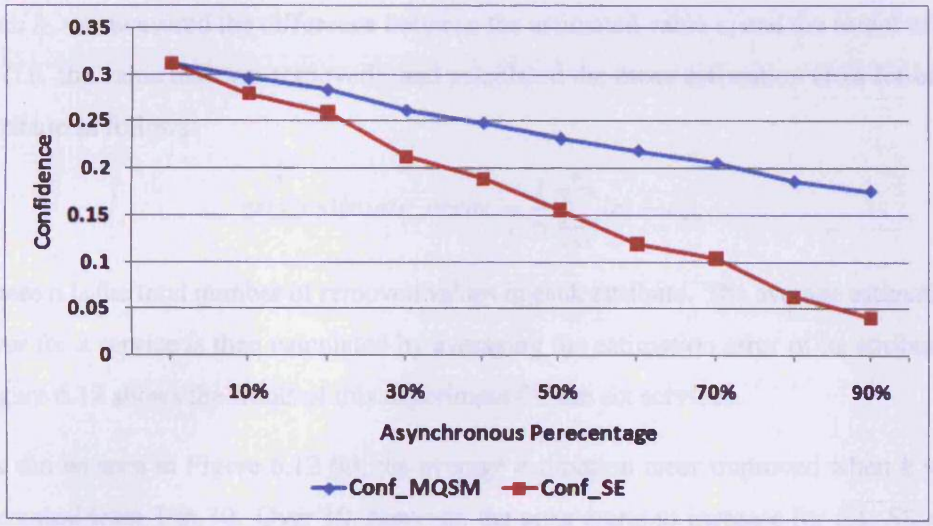


Figure 6.11: Impact of Asynchronous Data

As can be seen in Figure 6.11 the confidence measure of the assessment made by SE was significantly affected by the asynchronous data - 67% lower on average compared to MQSM. This is not surprising as SE uses only the synchronous subset of the data (i.e. only the paired $d(A_1)$ and $d(A_2)$), where as MQSM does not consider the need to pair the values in $d(A_1)$ and $d(A_2)$. So, the amount of data used by SE is significantly less

than the amount used by MQSM, hence it has a lower confidence value. The situation becomes worse as the percentage of asynchronous data increases.

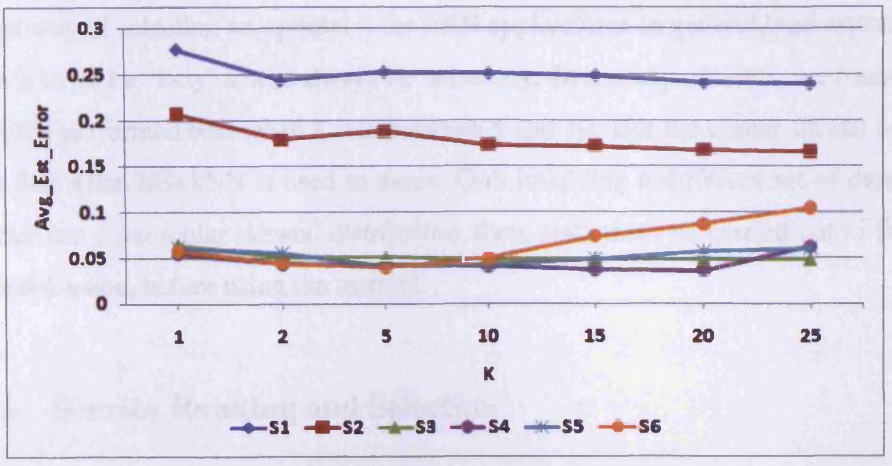
6.3.4 Quality of SE+kNN Assessment

SE+kNN relies on kNN to estimate the "unpaired" QoS data in assessment. It is known that the setting of k is critical to the performance of kNN and needs to be empirically observed. So in this experiment we study how the setting of k in SE+kNN may affect the performance of asynchronous data handling. We created two datasets containing 100 and 1000 tuples each, and we simulated asynchronous effect as before by removing 25% values from each attribute. We then run SE+kNN with k varying from 2 to 25. For each k, we measured the difference between the estimated value e_i and the actual value d_i (i.e. the value that was removed), and calculated the mean estimation error for each attribute as follows:

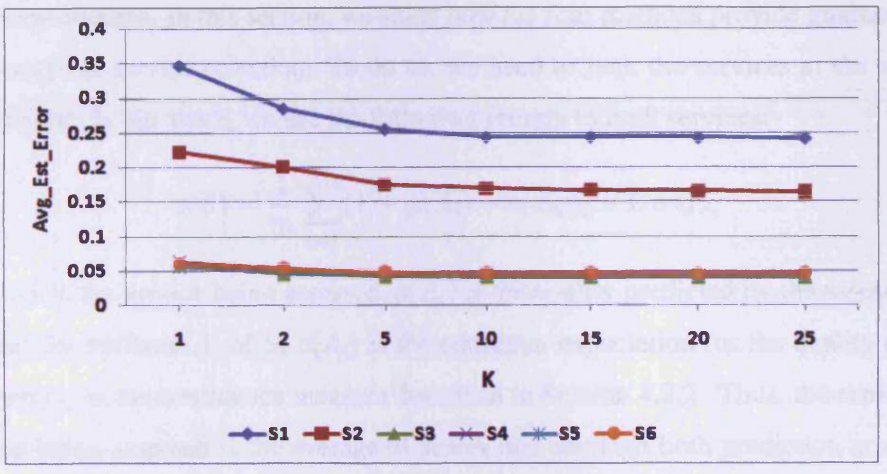
$$avg\_estimate\_error = \frac{1}{n}\sum_{i=1}^{n}|e_i - d_i|$$

where n is the total number of removed values in each attribute. The average estimation error for a service is then calculated by averaging the estimation error of its attributes. Figure 6.12 shows the result of this experiment for the six services.

As can be seen in Figure 6.12 (a), the average estimation error improved when k was increased from 1 to 10. Over 10, however, the error starts to increase for $S4$, $S5$ and $S6$. This is expected since for a single package service ($S3$), any number of neighbours selected will fall into the same package, hence a larger k should not affect the estimate. For $S4$, $S5$ and $S6$, a larger k implies the possibility that selected neighbours will actually fall into different packages, especially when the data size is relatively small. This can lead to estimate errors. This is evident in Figure 6.12 (a) as $S6$ has the worst estimate error due to the "narrowest" package band (0.3) it has. Figure 6.12 (b) adds further evidence to the preceding explanation. As the size of dataset increases, the chance of



(a) 100 observed instances



(b) 1000 observed instances

Figure 6.12: Estimation Error for Varied k

picking up neighbours from different packages reduces, even when the service package band is quite narrow. The high error for $S1$ and $S2$ is due the fact that the kNN method relies on synchronous data across the attributes to estimate asynchronous data, so a weak correlation between A_1 and A_2 for $S1$ and $S2$ can produce high error in estimation. Overall, the SE+kNN method can handle asynchronous data effectively when k is set around 5 to 10. However, it is worth emphasising that there is no theoretical or

proven way of selecting an optimal k for kNN applications in general, and experimental tests to find a "best" k will always be necessary. In our experiments, we found that SE+kNN performed best when k was between 5 and 10. But the reader should bear in mind that when SE+kNN is used to assess QoS involving a different set of data, e.g. one that has a particular skewed distribution, then, tests must be carried out to find an optimal k value, before using the method.

6.3.5 Service Ranking and Selection

The goal of QoS assessment is to help consumers to select a service that best meets their expectations. In this section, we study how the four methods provide guidance for consumers in service selection. To do so, we need to rank the services at the end of assessment. In our work, we use the following criteria to rank services:

$$r(S) = \frac{1}{m}\sum_{i=1}^{m}(1 - |p(A_i) - e(A_i)|) \times Conf_{A_i} \tag{6.1}$$

where S is the service being assessed, $p(A_i)$ is the quality predicted by the assessment method for attribute A_i of S, $e(A_i)$ is the consumer expectation for the quality of A_i, and $conf_{A_i}$ is the confidence measure described in Section 4.2.2. Thus, the rank for a service being assessed is the average of scores that combine both prediction accuracy ($|p(A_i) - e(A_i)|$) and confidence in that accuracy ($Conf_{A_i}$) for each attribute of the service. The highest ranked service is then recommended to the consumer.

Convergence Time

In this experiment, we study how quickly the assessment methods can converge to selecting the appropriate service [90]. We used the same datasets as before and removed 25% data from each attribute to simulate asynchronous data. We tested the four methods on the six services against $C8$'s request ($e(A_1) = 0.8, e(A_2) = 0.8$), with $k = 10$, $\lambda = 0.99$ and $\epsilon = 0.1$. The result of this experiment is shown in Figure 6.13.

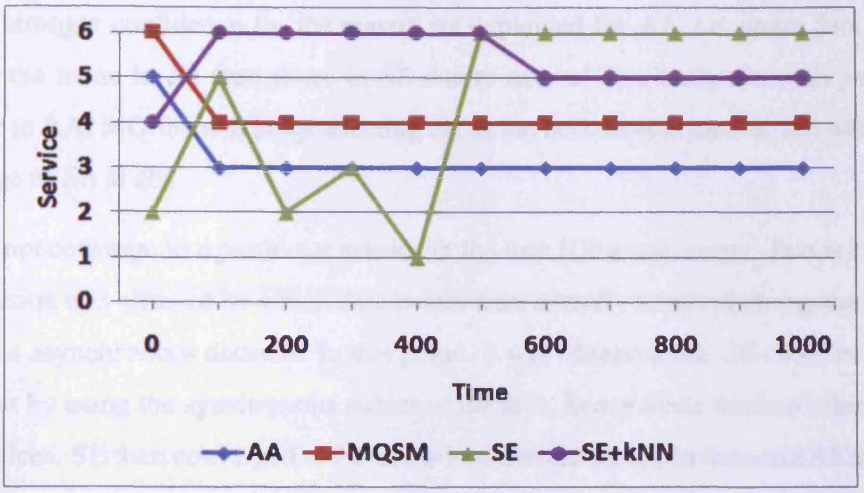


Figure 6.13: Convergence time for $C8$'s request

From Table 6.3, it is clear that $S5$ is the best choice for this request, so a good QoS assessment method should quickly converge to selecting $S5$ and then keep selecting it afterwards. As can be seen from Figure 6.13, AA ranked services wrong by choosing $S3$. This is because AA does not consider consumer expectation, nor service packages. As $S3$ offers a single package and due to normal distribution of data with a mean of 0.5 (see Table 6.3), it is easy to see that AA would average $S3$'s quality towards the mean with a large amount of data points around it. This would result in a strong confidence for the assessment (albeit a wrong one), and consequently $S3$ was ranked much higher over other services in all cases. For this reason, AA will keep selecting $S3$ as the best choice for $C8$ and not converge to $S5$ at all.

MQSM was mistakenly stuck with $S4$ as the best candidate for almost the same reason. Since MQSM considers consumer expectation in assessment, $S4$ and $S5$ would be identified by MQSM as being able to provide a quality nearer to what $C8$ has requested. However, MQSM does not consider quality packaging across attributes, so it did not recognise that the two 0.8 bands for A_1 and A_2 from $S4$ were in fact not attainable (see Table 6.3). As each attribute was assessed individually by MQSM, $S4$ would

have a stronger confidence for the reason we explained for AA, i.e. more data points around the mean in $S4$ than those in $S5$ due to normal distribution in each package. Similar to AA, MQSM will keep selecting $S4$ as the best service for $C8$, and will never converge to $S5$ at all.

SE did not converge to a particular service in the first 500 assessments. This is because its selection was affected by which data points were actually removed during the setting up of the asynchronous datasets. In this phase, it was observed that SE could not reach a verdict by using the synchronous subset of the data, hence made random selection of the services. SE then converged to $S6$ as the best service for $C8$ in the second half. This is because $S6$ was affected less by the asynchronous data than $S5$ was, so consequently it attained a higher ranking than $C5$ did due to a higher confidence score.

SE+kNN was picking up $S6$ initially as the best service for $C8$ for the same reason as SE was picking up $S6$ in the second half of the experiment. That is, $S5$ did not have a "large enough" synchronous subset to estimate the missing values in the asynchronous part effectively, whereas $S6$ was less affected by asynchronous data. After the first half, however, SE+kNN was able to converge to $S5$.

Since the convergence time for SE and SE+kNN is directly affected by which data points were removed during the setting up of the asynchronous datasets, we repeated this experiment many times and report the result in Figure 6.14.

As can be seen, the longest time (worst case) that SE+kNN took to converge to $S5$ was equivalent to the shortest time (best case) for SE to converge. On average, SE+kNN's converging time was 33% shorter than that of SE's. This provides clear evidence that by estimating missing values for asynchronous QoS data in assessment, our method can provide better guidance for consumers to choose their preferred services.

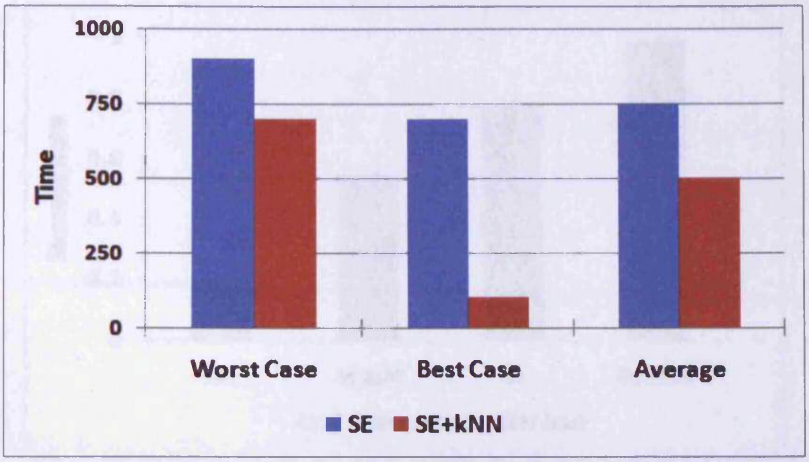


Figure 6.14: Convergence time for SE and SE+kNN

Selection Success Rate

In this experiment, we measure the rate of selecting a "correct" service by a QoS assessment. We used the same datasets and the same parameter settings described in the previous experiment, but assessed the six services given in Table 6.3 against all ten consumer requests defined in Table 6.4. The rate of successful selections is calculated as follows, where ϕ_c is the number of correctly selected services and ϕ is the total number of assessment requests:

$$Selection\ success\ rate = \frac{\phi_c}{\phi} \tag{6.2}$$

We ran the experiments five times and the average success rates for the four methods are shown in Figure 6.15. To explain the variation in success rate, we report the worst case (the lowest success rates achieved by the methods) in Figure 6.16.

As can be seen from Figure 6.16, AA selected $S3$ for all 10 assessment requests for the reason we gave in the previous experiment. It was correct only when the requester's expectation ($C1$ and $C9$) coincidentally met the qualities delivered by $S3$. Note that although the level of quality requested by $C1$ and $C9$ was also offered by $S2$, AA benefited from our confidence measure and recognised that the verdict on $S3$ was more

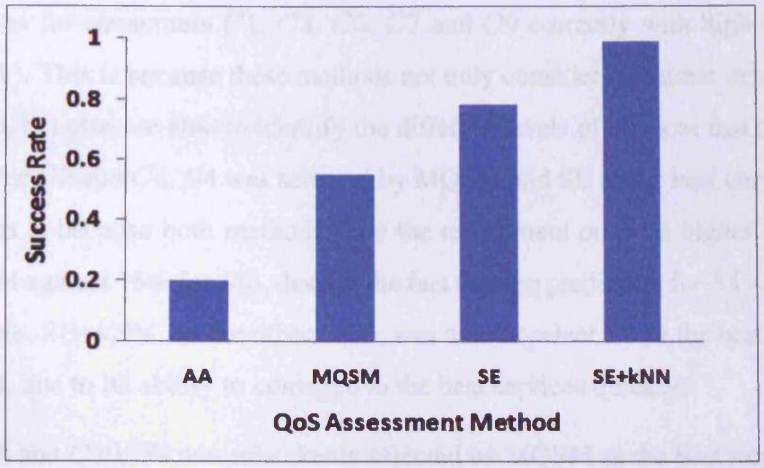


Figure 6.15: Selection success rates for the four methods

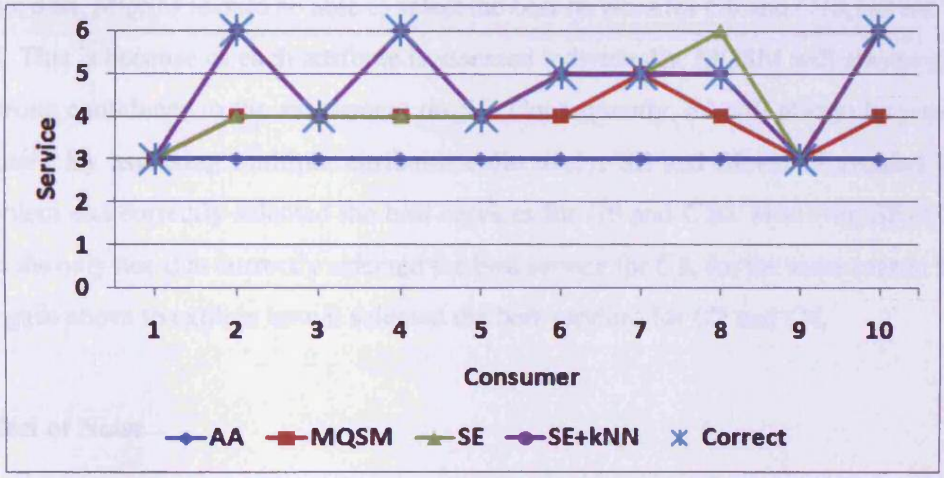


Figure 6.16: Worst case selections

reliable than that on $S2$. Our confidence measure has also led AA to avoid $S4$ which offers multiple levels of qualities, although $S4$'s average quality is the same as $S3$'s. Overall, since AA does not consider quality packages and consumer expectations, its selection success rate is pretty low.

The expectation-based methods (MQSM, SE and SE+kNN) were able to select the

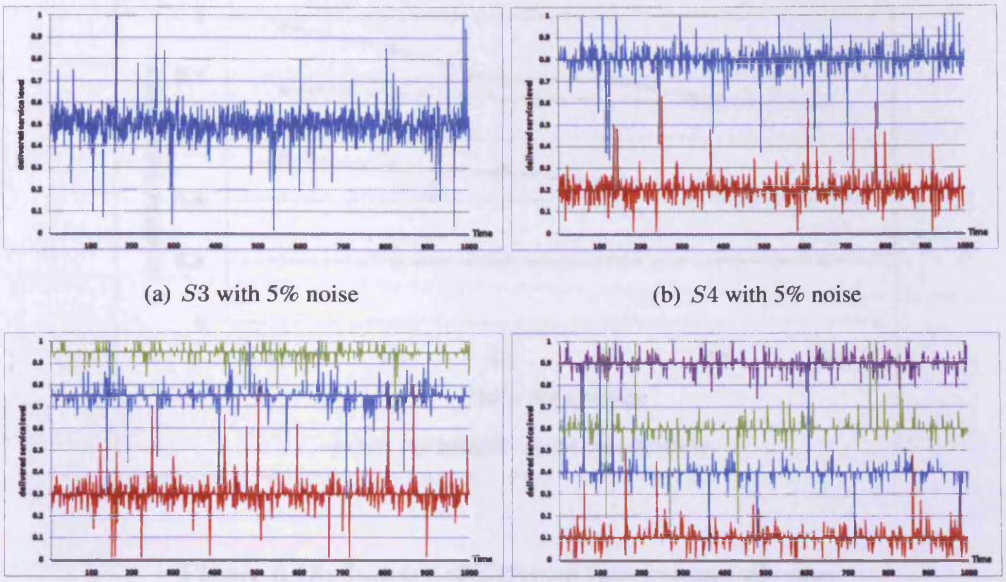
best services for consumers $C1$, $C3$, $C5$, $C7$ and $C9$ correctly with high confidence (about 95%). This is because these methods not only consider consumer expectation in assessment, but also are able to identify the different levels of services that $S4$, $S5$ and $S6$ offer. For $C2$ and $C4$, $S4$ was selected by MQSM and SE as the best choice instead of $S6$. This is because both methods gave the assessment on $S4$ a higher confidence (95% for $S4$ against 75% for $S6$), despite the fact that the prediction for $S4$ was actually less accurate. SE+kNN, on the other hand, was able to select $S6$ as the best choice for $C2$ and $C4$, due to its ability to converge to the best services quickly.

For $C6$, $C8$ and $C10$, $S4$ was mistakenly selected by MQSM as the best service. Since MQSM does not consider quality packaging across multiple attributes, it did not recognise that the two service levels delivered by $S4$ were in fact not attainable. By collecting more data, MQSM should be able to select the best services for $C6$ and $C10$, but not for $C8$. This is because as each attribute is assessed individually, MQSM will always give a strong confidence to the assessment on $S4$. Consequently, $S4$ will always be ranked higher. By assessing multiple attributes collectively, SE and SE+kNN avoided this problem and correctly selected the best services for $C6$ and $C10$. However, SE+kNN was the only one that correctly selected the best service for $C8$, for the same reason that we gave above to explain how it selected the best services for $C2$ and $C4$.

Effect of Noise

In the previous tests, our methods can be considered as being studied in an ideal situation. That is, each instance in the generated delivered QoS datasets is a true record of the delivered QoS. In practice, however, QoS monitoring process will be subject to various types of error [148]. In this section, we study how the four methods perform when datasets are not perfect but contain erroneous instances, thereby showing the robustness of each method in dealing with noise.

To do so, we introduce noises into service delivery data. We generated random noises from a given range and added them to delivered qualities. The level and percentage of



(a) $S3$ with 5% noise

(b) $S4$ with 5% noise

(c) $S5$ with 5% noise

(d) $S6$ with 5% noise

Figure 6.17: Delivered qualities for A_1 with noises

noises were controlled to simulate a range of situations, from those where there is still a clear separation between different quality packages after the noises being added, to those where the noises would result in a substantial overlapping between the packages. Figure 6.17 shows the delivered qualities by $S4$, $S5$ and $S6$ with 5% random noises from the range [0.05, 0.5] added.

In the first experiment, we tested the effect of different percentages of noises on selection success rate for the four methods. We used the same dataset described in the previous experiments, but introduced a noise of 0.25 randomly into $d(A_1)$ and $d(A_2)$, and varied the percentage of noises from 2% to 10%. As previously, we assessed the six services against the ten consumer requests. The result (averaged over five runs) is shown in Figure 6.18.

As can be seen in Figure 6.18, different percentages of noise introduced had not deviated the selection success rate for AA. This is expected since even with 10% noise, the

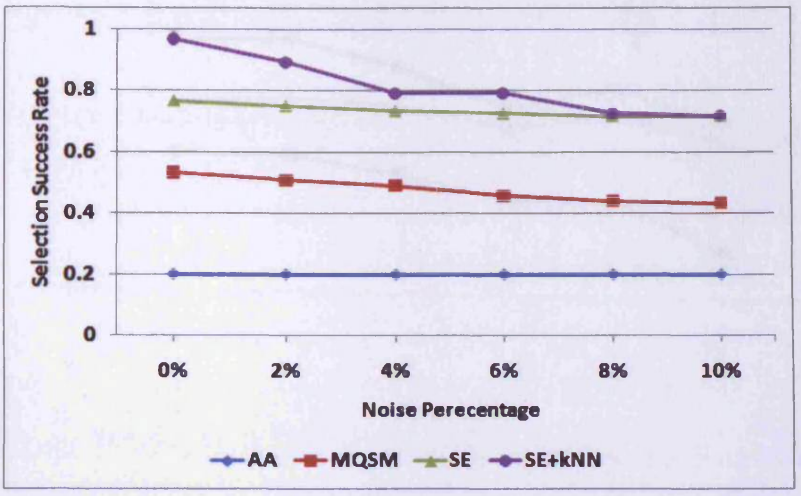


Figure 6.18: Impact of different percentages of noise

rest of the data (90%) for $S3$ are still close enough to the mean, resulting in a strong confidence for the assessment of $S3$ by AA over other services in all cases. The result for the expectation-based methods (MQSM, SE and SE+kNN), on the other hand, showed a clear decrease in selection success rate as the amount of noise added to the data increased. Compared to MQSM and SE+kNN, the success rate for SE was more stable. This may be attributed to the collective assessment of the two attributes by SE: the noise introduced for one attribute would make the instance concerned no longer meet the consumer expectation, hence effectively ignored by SE. Although SE+kNN assessed A_1 and A_2 collectively too, its selection success rate decreased. This is because the noise introduced did affect the accuracy of kNN in handling asynchronous data, which then affected selection success rate.

In the second experiment, we tested the success selection rate for the four methods on the delivered data with different levels of noise introduced. We used the same dataset that was used in the previous experiment, but this time we had the percentage of noise fixed at 5% and varied the level of noise from 0.1 to 0.5. The result is shown in Figure 6.19.

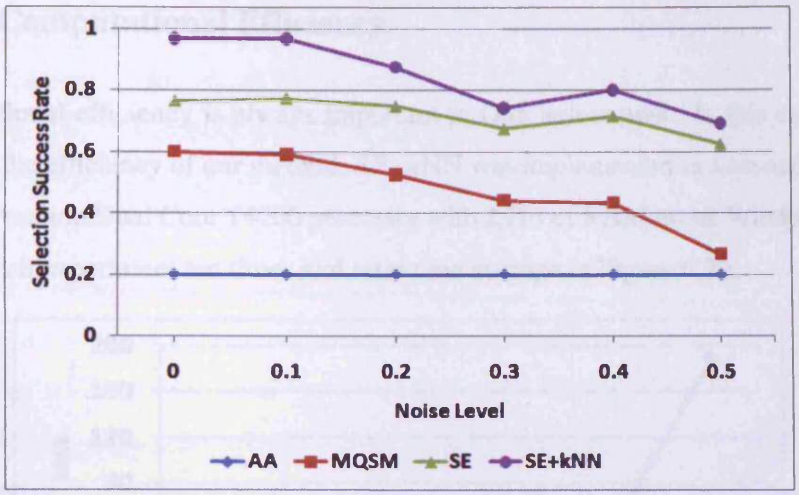


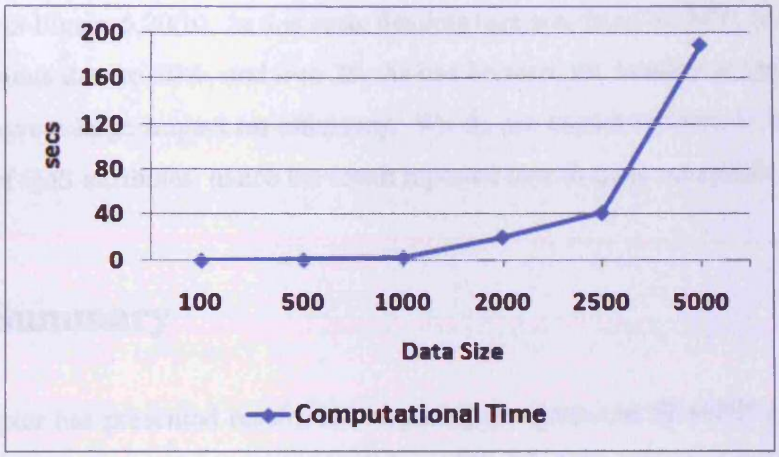
Figure 6.19: Impact of different levels of noise

It is easy to observe that the patterns shown in Figure 6.19 are largely the same as those shown in Figure 6.18, so the effects of percentage and level of noise on the four methods are similar. Relatively more substantial degradation on success rate occurred when the level of noise was over 0.25, due to the fact that above this point the data from different service packages began to exhibit a high degree of overlapping (see Figure 6.17). SE and SE+kNN are still more robust than MQSM in this experiment: the selection success rate for MQSM was dropped by 55% when the level of noise was 0.5, whereas SE and SE+kNN only dropped their rates by 18% and 28%, respectively, for the same level of noise. This is largely due to their collective assessment of multiple attributes.

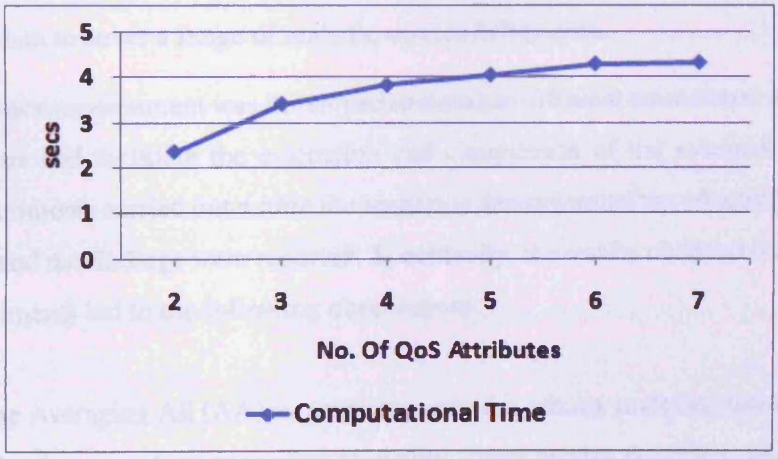
Overall, as expected, noise in QoS data has led to a general decrease in selection success rate. However, our method has shown more tolerance to noise and given relatively higher selection success rate than other methods. It is evident that our proposed method has coped reasonably well even when the level of noise was 0.5 and the percentage of noise reached 10%, which resulted in overlappings between packages that we do not expect to encounter normally in real world applications. We therefore expect our proposed method to perform well in practice.

6.3.6 Computational Efficiency

Computational efficiency is always important to QoS assessment. In this experiment we study the efficiency of our method. SE+kNN was implemented in Java and ran on a 2.0GHz Pentium Dual Core T4200 processor with 2 GB of RAM under Windows Vista. We ran each experiment ten times and report the average in Figure 6.20.



(a) Runtime vs Data Size



(b) Runtime vs No. QoS Attributes

Figure 6.20: Computational Efficiency of SE+kNN

In Figure 6.20(a), we report the time taken by SE+kNN to process data of various sizes, while fixing the number of attributes to two, percentage of asynchronous data to 50% and k to 25. The kNN method has a complexity of $O(n^2)$, where n is the number of data points, so the trend in time was expected. A possible improvement would be to consider a *windowed* approach, where kNN search is confined to a specified time window. Also, we tested the computational time against the number of QoS attributes and the result is shown in Figure 6.20(b). In this case, the data size was fixed to 1000, percentage of asynchronous data to 50%, and k to 25. As can be seen, the number of QoS attributes did not have a large impact on efficiency. We do not expect services to have a huge number of QoS attributes, hence the result reported here is quite acceptable.

6.4 Summary

This chapter has presented results of comparing our proposed SE+kNN against three other representative methods (AA, MQSM and SE). We defined a set of criteria for measuring the effectiveness of the methods under study, and we designed a set of scenarios and test data to cover a range of realistic service behaviours.

A simulation environment was developed to simulate different consumers' and providers' behaviours and facilitate the evaluation and comparison of the assessment methods. The experiments carried out within the scenarios demonstrated the effectiveness of each method and the findings were reported. Specifically, the results obtained from the range of experiments led to the following observations:

- The Averaging All (AA) assessment method is wholly inappropriate for use when adopting a conformance view of quality where service providers deliver multiple levels of services to different consumers based on their expectations. In such cases, the AA method produced inaccurate predictions and misled consumers by nominating unsuitable services in selection.

- The Multiple Quality Space Mapping (MQSM) and Synchronous Extension (SE) assessment methods are both sensitive to properties relating to consumer behaviours. When a consumer expressed expectation on a single attribute, they showed good performance in terms of assessment accuracy and confidence. However, when the consumer's request involved multiple attributes, neither MQSM nor SE dealt with it adequately and showed a significant decrease in performance. While the MQSM method produced inaccurate assessment, the SE method showed a significant decrease in confidence when the qualities of the multiple attributes were monitored asynchronously.

- The SE+kNN method, which used the kNN algorithm to prepare asynchronous data before making assessment, was able to overcome the weaknesses of the MQSM and SE methods in handling multiple attributes. This method was able to give better performance for QoS assessment over multiple attributes with asynchronous data than its counterparts did.

- The average estimation error for the kNN algorithm to prepare asynchronous data decreased as the size of the QoS dataset increased. This improved QoS assessment. However, increasing the size of the QoS dataset meant the kNN algorithm took longer to prepare the asynchronous data. Since we envisage that data preparation will be carried out off-line before making assessment, this is considered acceptable. The experiment showed that the number of QoS attributes, however, did not have a large impact on efficiency.

- Having obtained asynchronous data by means of the kNN algorithm, the SE+kNN method was able to converge faster than its counterparts to select the right services. This implies that our proposed method could rank a set of alternative services more effectively, hence giving consumers better guidance by providing the information that is needed to enable selection of the appropriate service among the alternatives (i.e. the one that can best meet their requirements).

- Both data size and data deviation were established as being appropriate measures

of confidence. The confidence verdict produced from these measures was able to reliably recognise situations where the prediction accuracy of the QoS assessment method was affected and provided lower confidence in such situations.

- The SE+kNN method was effective in environments where the dynamic behaviours of service providers and consumers were assumed. This was demonstrated by achieving a higher selection success rate using the SE+kNN method than the other methods did in different scenarios.

- The SE+kNN method was fairly robust in response to noise. The range and level of noise introduced in the experiments did not significantly change the selection success rate of the SE+kNN method. This provided evidence that the method can be used in real-world applications where noise may exist in QoS data collection.

Through conducting the above evaluation, it has been shown that the expectation-based methods (MQSM, SE and SE+kNN) provide significant improvement over the Averaging All (AA) assessment method, which cannot operate effectively when the conformance view of quality is adopted. The MQSM, SE and SE+kNN assessment methods are shown to be effective tools for supporting consumers' decision-making in finding services which best meet their needs in a service provisioning environment. However, the performance of the MQSM and SE methods depends upon a specific condition, that is, their performance (accuracy and confidence) is less reliable when consumer expectations are over multiple attributes, service providers offer multiple packages, and the QoS data is asynchronously collected by monitoring tools. Our proposed method, SE+kNN, overcomes these limitations and successfully achieves high performance in QoS assessment over multiple attributes.

Chapter 7

Conclusions and Future Work

This chapter concludes the research reported in this thesis, which was undertaken to enable the quality of service to be more effectively assessed to support consumers when choosing their preferred services. More specifically, this research reviewed and evaluated some representative QoS assessment methods in handling multiple attributes, especially when the qualities of these attributes are assumed to be monitored asynchronously. Having described a number of limitations of these methods, a new QoS assessment method was developed to deal more effectively with multiple attributes. A k-nearest neighbour (kNN) technique was employed to transform asynchronous data to a synchronous form before using it in assessment. By using kNN to handle asynchronous data, we are able to use the data collected more effectively and hence improve assessment confidence.

This Chapter is structured as follows. Section 7.1 reviews the contributions of this research and Section 7.2 discusses the main ways in which this research can be carried forward in the future.

7.1 Research Contributions

This thesis has presented the SE+kNN method, an extension of MQSM method, for assessing quality of a service in an open and dynamic service provisioning environment. Before summarising SE+kNN's contributions to the state of the art, we recap the re-

quirements that must be taken into account in order to develop a good QoS assessment method (Section 2.3).

- **R1:** Service providers should be assessed per consumer request (or requirement), because different requesters may have different quality requirements, and a service suitable for one consumer may not be suitable for another.
- **R2:** In producing QoS verdicts, QoS assessment should take into account:
  - **a)** The dynamicity of service behaviour over time.
  - **b)** Contextual information that leads to the provision of a certain level of quality.
  - **c)** Multiple levels of quality offered by a single service.
- **R3:** The confidence of an QoS assessment is provided for every QoS verdict.
- **R4:** QoS assessment should deal with multiple attributes collectively.

From our investigation in Chapter 2, we found that no QoS assessment method exists that satisfies all these requirements. In what follows, we will show how these requirements are met by the SE+kNN method and highlight its novelty.

A generic QoS assessment process was proposed in Chapter 3 and was used to understand what is involved in the process of QoS assessment. In this model, QoS assessment is viewed as involving four fundamental components: data collection, data selection, data aggregation, and service ranking. In the following we describe how SE+kNN was built w.r.t. each component and how these contributed to satisfying the aforementioned requirements.

The concern of *data collection* focuses on what and how data relevant to the quality of a service may be obtained. Broadly, there are three types of data: advertisements, user ratings and monitored QoS data. SE+kNN used monitored QoS data as the basis for assessment. By using the monitored data, SE+kNN was able to capture the dynamic

behaviour of a service provider over time (**R2a**). In addition, user expectations were collected alongside QoS data. These expectations may be inferred from a Service Level Agreement (SLA), which is a formal contract between service providers and consumers before using the service. SE+kNN used user expectations in assessment. By using this information SE+kNN was able to identify and recognise the reason behind delivering a certain level of service to consumers and hence satisfied **R2b**.

The concern of *data selection* is about which data should be selected and used in assessment. There are different mechanisms applied for data selection in the literature. A simple one is to consider all data as relevant. The Averaging All method is as an example of this approach. More advanced techniques use various heuristics, such as expectation and collaborative filtering. SE+kNN is an extension of the MQSM method which employs expectation-based data selection. That is, consumers are asked to state their preferred service levels as part of their assessment requests, and only the data that is similar to their expectations will be selected and used in assessment. Thus, SE+kNN inherits the good characteristics from MQSM in identifying possible multiple levels of quality delivered by a single service (**R2c**). By selecting only the data that is relevant to the assessment request, SE+kNN was able to assess a service per request and hence recognise the most suitable service for the requester (**R1**).

In the *data aggregation* task, the selected QoS data is aggregated to indicate a QoS level that the service provider is likely to deliver. For multiple attributes, in contrast to MQSM, SE+kNN selects and aggregates multiple attributes collectively rather than individually. Thus, SE+kNN was able to satisfy **R4** by being able to deal with multiple attributes adequately.

In addition to the three components described above, we developed two more techniques in our research: a confidence computational model and data preparation to make asynchronous data usable. The former is used to indicate how likely the derived assessment verdict is to be useful. To do so, we used two measures, data size and deviation. A confidence verdict is then established using these two measures which indicates how

reliable a QoS verdict is (**R3**). This confidence model was integrated into the QoS assessment process (see Figure 3.1) to help service ranking and consequently help consumers make better decisions in service selection. It is worth noting that our proposed confidence model is generic. That is, it is not limited to SE+kNN, but can be integrated into other QoS assessment methods (e.g, Averaging All and MQSM methods) too.

Since SE+kNN assesses multiple attributes collectively, this limits its ability to use all available data. The adjustment made by SE to deal with multiple attributes implies that QoS data for multiple attributes is assumed to be synchronously collected. To address this issue, we prepared the collected QoS data by transforming asynchronous data to a synchronous form. The asynchronous data was treated as a dataset containing "missing" values and a kNN based technique was used to estimate the missing ones. Having processed asynchronous data, SE+kNN was able to improve the confidence of assessment and make better service selection for consumers, especially with multiple attributes (**R3**).

This research makes the following contributions to the state of the art:

- A conceptual model for QoS assessment: An abstract model for QoS assessment was given to enable us to understand what is involved in the QoS assessment process. It was used to describe and contrast approaches to QoS assessment. This model was then used to guide the development of SE+kNN for QoS assessment described in this thesis (Chapter 3).

- A probabilistic model to quantify confidence in QoS Assessment: This model produces a reliability value for each QoS assessment result to determine the validity of assessment. This value is calculated based on the characteristics of the QoS data used in the assessment. More specifically, this value is produced by integrating two reliability measures: the number of QoS data items used in the assessment and the variation of data in the dataset (Chapter 4).

- Handling asynchronous data: Asynchronous data were treated as a dataset con-

taining "missing" values and a kNN based technique was used to estimate the missing ones. This enabled us to use the data more effectively when assessing a service based on multiple attributes (Chapter 5).

In order to verify our claims, empirical evaluation was carried out and it was demonstrated that:

- SE+kNN is able to not only assess a service per request more accurately, but also with high confidence. This was demonstrated by the performance of SE+kNN to handle various behaviours of consumers, providers and data (Section 6.3.2).
- In producing QoS verdicts, SE+kNN was able to capture the dynamicity of service providers and consumers. This was confirmed by the selection success rate that was achieved by SE+kNN in service selection (Section 6.3.5).
- The data preparation component developed in this research was shown to play an important role in SE+kNN operation and significantly contribute to its overall performance. By integrating data preparation into the QoS assessment process, SE+kNN is able to handle asynchronous data and get the most out of collected data, which consequently improves the result obtained from SE+kNN. This was demonstrated by the short time that the SE+kNN took to converge to the most suitable service for a requester compared to its counterparts (Section 6.3.5).
- The confidence model developed in this research was used to give an indication of the reliability of QoS verdicts. This component is generic as it can be integrated into any assessment method. This was verified by using it with different assessment methods. For example, the Averaging All method benefited from using the confidence model to differentiate between a consistent and fluctuating service provider in service ranking and selection (Section 6.3.5).

To sum up, SE+kNN satisfies all the above requirements (**R1-R4**) for a QoS assessment method. By satisfying all these requirements our proposed method was able to help fill

'alignment gap', 'execution gap', and 'perception gap' in the quality cycle in Figure 2.1. Considering consumers' expectations in QoS assessment made SE+kNN able to fill the gap between consumers' requirements and services' offers. The 'execution gap' has been covered by using monitoring data in assessment. This made SE+kNN able to capture any variation between what was advertised and delivered by service providers. Finally, by adopting conformance view of quality, SE+kNN was able to consider the 'perception gap' which may arise from having different expectations on quality of a service by consumers. The conducted empirical evaluation indicated that SE+kNN is suitable for use in real world contexts.

7.2 Future Work

As highlighted in the previous section, this research makes a number of contributions to the state of the art. However, there are still a number of ways in which the work can be further extended. In particular, we identify the following two areas in which further significant research is warranted.

7.2.1 Model Based Approaches

The QoS assessment methods reviewed and developed in this research are all dependent on the instance-based approach. That is, these methods carry out data selection and aggregation each time an assessment is requested by a particular consumer. For the SE+kNN method, this involves a degree of computational complexity that is higher than other methods (e.g. Averaging All and MQSM) with respect to time since it uses the kNN algorithm to handle asynchronous data. This problem is compounded by increases in the number of attributes and data points, as we discussed in our experimental study. One way in which the complexity might be reduced involves taking a model-based approach to assessment. In such an approach, rather than using all historic service

performance data as points, a mathematical model of behaviour may be created and maintained for each service. Such a model would be consulted during each instance of assessment, and updated when new information became available. Investigation of the use of the model-based approach [104, 144] to handle multiple attributes should be considered.

7.2.2 Multiple Criteria Decision-Making

One potential extension for our work is to consider how services may be ranked based on multiple attributes. The approach to service ranking used in this research was to combine the outcome of assessment, accuracy and confidence, into a single verdict for each service under assessment, and then rank the services based on their numerical order. Unfortunately, it is not always desirable or possible to derive a single verdict, for example, when the quality of each attribute must be considered and compared separately. In addition, service ranking using a single verdict derived from multiple criteria could miss consumer desired services. Consider Alice as an example again, suppose that Alice wishes to find a web hosting service that can deliver 800 requests per second. Assume that there are two candidate services $S1$ and $S2$. Assume also that the outcome of assessments for the two candidate services are: $S1$:$<$ 800 requests/second, $0.75 >$ and $S2$:$<$ 750 requests/second, $0.90 >$ in terms of accuracy and confidence, respectively. In other words, $S1$ is considered to be able to offer a better service for Alice (i.e. closer to the level requested by Alice) but with low confidence, while $S2$ is predicted to deliver less quality but with high confidence. In this case, it is hard to decide which service is better for Alice. Expressing the QoS as a single verdict would hide this information which can be useful and supportive in service selection and ranking. To overcome this issue, more sophisticated solutions based on multiple criteria decision-making principles [34] need to be considered in service ranking. This extension would greatly enhance the performance of our proposed method, SE+kNN, in terms of its customisability, since it would allow a wide range of criteria to be encoded

and used in service selection.

Refereed Papers

Large portions of Chapters 3, 4, 5, and 6 have been presented in the proceedings of peer reviewed International Conferences and Journals.

1. H. Al-Dossari, A. Preece, J. Shao. Improving QoS Assessment Involving Multiple Attributes. In 26th British National Conference on Databases PhD Forum (BNCOD 2009). Birmingham, 2009.

2. H. Al-Dossari, J. Shao, A. Preece. QoS Assessment over Multiple Attributes. In 10th International Conference on Computer and Information Technology (CIT 2010), pp 1456-1461. IEEE, 2010.

3. H. Al-Dossari, J. Shao, A. Preece. Handling Asynchronous Data in Assessing QoS over Multiple Attributes. In International Conference on Service Oriented Computing and Application (SOCA 2010), pp 1-8. IEEE, 2010.

4. H. Al-Dossari, J. Shao, A. Preece. Improving QoS Assessment over Multiple Attributes with Asynchronous Data. Accepted to be published in International Journal of Computer Systems Science and Engineering (IJCSSE). CRL Publisher.

Bibliography

[1] eBay. http://www.ebay.com. 20 May 2011.

[2] NetBeans. http://netbeans.org/index.html. 20 May 2011.

[3] E. Acuna and C. Rodriguez. The treatment of missing values and its effect in the classifier accuracy. pages 639–647. Springer, 2004.

[4] M. Aiello, G. Frankova, and D. Malfatti. What's in an Agreement? An Analysis and an Extension of WS-Agreement. In *3rd International Conference on Service-Oriented Computing (ICSOC 2005)*, pages 424–436. Springer, 2005.

[5] F. Barbon, P. Traverso, M. Pistore, and M. Trainotti. Run-time monitoring of instances and classes of web service compositions. In *4th International Conference on Web Services (ICWS 2006)*, pages 63–71. IEEE Computer Society, 2006.

[6] A. Basu, I. Cheng, and Y. Yu. Multi-Server optimal bandwidth monitoring for QoS based multimedia delivery. In *the 2003 International Symposium on Circuits and Systems (ISCAS 2003)*, volume 2, pages 812–815. IEEE, 2003.

[7] G. Batista and M.C. Monard. A study of K-nearest neighbour as a model-based method to treat missing data. In *Argentine Symposium on Artificial Intelligence*. LABIC - Laboratory of Computational Intelligence, 2001.

[8] G.E. Batista and M.C. Monard. An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17(5):519–533, 2003.

[9] M. Bechler, H. Ritter, and J.H. Schiller. Quality of service in mobile and wireless networks: The need for proactive and adaptive applications. In *33rd Annual Hawaii International Conference on System Sciences (HICSS 2000)*, pages 10–19. IEEE, 2000.

[10] N. Bhatti, A. Bouch, and A. Kuchinsky. Integrating user-perceived quality into web server design. *Computer Networks*, 33(1-6):1–16, 2000.

[11] M. Blankers, M.W.J. Koeter, and G.M. Schippers. Missing data approaches in eHealth research: Simulation study and a tutorial for nonmathematically inclined researchers. *Journal of Medical Internet Research*, 12(5), 2010.

[12] Z. Bodó and Z. Minier. On supervised and semi-supervised k-nearest neighbor algorithms. In *7th Joint Conference on Mathematics and Computer Science*, volume 53, pages 79–92. INFORMATICA, 2008.

[13] A. Bouch, A. Kuchinsky, and N. Bhatti. Quality is in the eye of the beholder: meeting users' requirements for Internet quality of service. In *SIGCHI conference on Human factors in computing systems*, pages 297–304. ACM, 2000.

[14] L.P. Bras and J.C. Menezes. Improving cluster-based missing value estimation of DNA microarray data. *Biomolecular engineering*, 24(2):273–282, 2007.

[15] M.L. Brown and J.F. Kros. Data mining and the impact of missing data. *Industrial Management & Data Systems*, 103(8):611–621, 2003.

[16] S.F. Buck. A method of estimation of missing values in multivariate data suitable for use with an electronic computer. *Journal of the Royal Statistical Society. Series B (Methodological)*, 22(2):302–306, 1960.

[17] G. Canfora and M. Di Penta. Testing services and service-centric systems: Challenges and opportunities. *IT Professional*, 8(2):10–17, 2006.

[18] MH Cartwright, MJ Shepperd, and Q. Song. Dealing with missing software project data. In *9th International Software Metrics Symposium*, pages 154–164. IEEE Computer Society, 2003.

[19] S. Chee, J. Han, and K. Wang. Rectree: An efficient collaborative filtering method. In *Data Warehousing and Knowledge Discovery*, pages 141–151. Springer, 2001.

[20] J. Chen and J. Shao. Nearest neighbor imputation for survey data. *Journal of Official Statistics - Stockholm*, 16(2):113–132, 2000.

[21] X. Chen, X. Liu, Z. Huang, and H. Sun. RegionKNN: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation. In *8th International Conference on Web Services (ICWS 2010)*, pages 9–16. IEEE Computer Society, 2010.

[22] Z. Chen, C. Liang-Tien, B. Silverajan, and L. Bu-Sung. UX-an architecture providing QoS-aware and federated support for UDDI. In *1st International Conference on Web Services (ICWS 2003)*, pages 171–176. IEEE Computer Society, 2003.

[23] R.J. Cook, L. Zeng, and G.Y. Yi. Marginal analysis of incomplete longitudinal binary data: a cautionary note on LOCF imputation. *Biometrics*, 60(3):820–828, 2004.

[24] A. Dan, H. Ludwig, G. Pacifici, et al. Web service differentiation with service level agreements. *White Paper, IBM Corporation*, 2003.

[25] C. Dellarocas. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In *2nd International Conference on Electronic Commerce (ICEC 2000)*, pages 150–157. ACM, 2000.

[26] Rubin D. Dempster, A. Incomplete data in sample surveys. In *in Madow, W.G., Olkin, I., Rubin, D. (Eds),Sample Surveys Vol. II: Theory and Annotated Bibliography*, pages 3–10. Academic Press, 1983.

[27] V. Deora, J. Shao, W.A. Gray, and N.J. Fiddian. A quality of service management framework based on user expectations. In *1st International Conference on Service-Oriented Computing (ICSOC 2003)*, pages 104–114. Springer, 2003.

[28] P. Domingos. Unifying instance-based and rule-based induction. *Machine Learning*, 24(2):141–168, 1996.

[29] K. Driessens and S. Džeroski. Integrating guidance into relational reinforcement learning. *Machine Learning*, 57(3):271–304, 2004.

[30] K. Driessens and S. Džeroski. Combining model-based and instance-based learning for first order regression. In *22nd International Conference on Machine Learning (ICML 2005)*, pages 193–200. ACM, 2005.

[31] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*, volume 2. Citeseer, 2001.

[32] A.J. Elliot and J.W. Fryer. The goal construct in psychology. *Handbook of motivation science*, pages 235–250, 2008.

[33] A. Erradi, S. Padmanabhuni, and N. Varadharajan. Differential QoS support in web services management. In *4th International Conference on Web Services (ICWS 2006)*, pages 781–788. IEEE Computer Society, 2006.

[34] J. Figueira, S. Greco, and M. Ehrgott. *Multiple criteria decision analysis: state of the art surveys*. Springer Verlag, 2005.

[35] D. Gambetta. Can we trust trust. *Trust: Making and Breaking Cooperative Relations, electronic edition, Department of Sociology, University of Oxford*, pages 213–237, 2000.

[36] P.J. García-Laencina, J.L. Sancho-Gómez, A.R. Figueiras-Vidal, and M. Verleysen. K nearest neighbours with mutual information for simultaneous classification and missing data imputation. *Neurocomputing*, 72(7-9):1483–1493, 2009.

[37] G. Gmel. Imputation of missing values in the case of a múltiple item instrument measuring alcohol consumption. *Statistics in medicine*, 20(15):2369–2381, 2001.

[38] D. Gouscos, M. Kalikakis, and P. Georgiadis. An approach to modeling web service qos and provision price. In *4th International Conference on Web Information Systems Engineering Workshops (WISEW 2003)*, pages 121–130. IEEE, 2003.

[39] J. Grzymala-Busse and M. Hu. A comparison of several approaches to missing attribute values in data mining. In *Rough sets and current trends in computing*, pages 378–385. Springer, 2001.

[40] S. Guha, D. Gunopulos, and N. Koudas. Correlating synchronous and asynchronous data streams. In *9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003)*, pages 529–534. ACM, 2003.

[41] J.F. Hair, W.C. Black, B.J. Babin, R.E. Anderson, and R.L. Tatham. *Multivariate data analysis*, volume 6. Prentice hall Upper Saddle River, NJ, 2006.

[42] E. Hajizadeh, H. Davari, Ardakani, and J. Shahrabi. Application of data mining techniques in stock markets: A survey. *Journal of Economics and International Finance*, 2(7):109–118, 2010.

[43] R.M. Hamer and P.M. Simpson. Last observation carried forward versus mixed models in the analysis of psychiatric clinical trials. *American Journal of Psychiatry*, 166(6):639, 2009.

[44] L. Harada. An efficient sliding window algorithm for detection of sequential patterns. In *8th International Conference on Database Systems for Advanced Applications (DASFAA 2003)*, pages 73–80. IEEE Computer Society, 2003.

[45] J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237. ACM, 1999.

[46] J.L. Herlocker, J.A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *the 2000 ACM Conference on Computer Supported Cooperative Work*, pages 241–250. ACM, 2000.

[47] J.L. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

[48] M.A. Hernández and S.J. Stolfo. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data mining and knowledge discovery*, 2(1):9–37, 1998.

[49] CC Holmes and NM Adams. A probabilistic nearest neighbour method for statistical pattern recognition. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(2):295–306, 2002.

[50] P.R. Houck, S. Mazumdar, T. Koru-Sengul, G. Tang, B.H. Mulsant, B.G. Pollock, and C.F. Reynolds III. Estimating treatment effects from longitudinal clinical trial data with missing values: comparative analyses using different methods. *Psychiatry research*, 129(2):209–215, 2004.

[51] D.C. Howell. Treatment of missing data. *The Sage handbook of social science methodology*, pages 208–224, 2007.

[52] Z. Huang, H. Chen, and D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):116–142, 2004.

[53] M. Huisman. Imputation of missing item responses: Some simple techniques. *Quality and Quantity*, 34(4):331–351, 2000.

[54] T.D. Huynh, N.R. Jennings, and N.R. Shadbolt. FIRE: An integrated trust and reputation model for open multi-agent systems. In *16th European Conference on Artificial Intelligence*, volume 16, pages 18–21. IOS Press, 2004.

[55] T.D. Huynh, N.R. Jennings, and N.R. Shadbolt. An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 13(2):119–154, 2006.

[56] J.M. Jerez, I. Molina, P.J. García-Laencina, E. Alba, N. Ribelles, M. Martín, and L. Franco. Missing data imputation using statistical and machine learning methods in a real breast cancer problem. *Artificial Intelligence in Medicine*, 50(2):105–115, 2010.

[57] P. Jönsson and C. Wohlin. Benchmarking k-nearest neighbour imputation with homogeneous Likert data. *Empirical Software Engineering*, 11(3):463–489, 2006.

[58] A. Jøsang and J. Haller. Dirichlet reputation systems. In *2nd International Conference on Availability, Reliability and Security (ARES 2007)*, pages 112–119. IEEE Computer Society, 2007.

[59] A. Jøsang and R. Ismail. The beta reputation system. In *15th Bled Electronic Commerce Conference (BECC 2002)*, volume 160. Citeseer, 2002.

[60] A. Jøsang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.

[61] A. Jøsang, X. Luo, and X. Chen. Continuous ratings in discrete bayesian reputation systems. In *Trust Management II*, pages 151–166. Springer, 2008.

[62] A. Jøsang and W. Quattrociocchi. Advanced features in Bayesian reputation systems. In *Trust, Privacy and Security in Digital Business*, pages 105–114. Springer, 2009.

[63] I.J. Jureta, C. Herssens, and S. Faulkner. A comprehensive quality model for service-oriented systems. In *Software Quality Journal*, volume 17, pages 65–98. Springer, 2009.

[64] S. Kalepu, S. Krishnaswamy, and S.W. Loke. Reputation= f (user ranking, compliance, verity). In *2nd International Conference on Web Services (ICWS 2004)*, pages 200–207. Published by the IEEE Computer Society, 2004.

[65] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *12th International Conference on World Wide Web (WWW 2003)*, pages 640–651. ACM, 2003.

[66] A. Keller and H. Ludwig. Defining and monitoring service level agreements for dynamic e-business. In *16th USENIX System Administration Conference*. The USENIX Association, 2002.

[67] A. Keller and H. Ludwig. The WSLA framework: Specifying and monitoring service level agreements for web services. *Journal of Network and Systems Management*, 11(1):57–81, 2003.

[68] S.N.L.C. Keung and N. Griffiths. Using recency and relevance to assess trust and reputation. In *Proceedings of AISB 2008 Symposium on Behaviour Regulation.* The Society for the Study of Artificial Intelligence and Simulation of Behaviour, 2008.

[69] D. Kim, S. Lee, S. Han, and A. Abraham. Improving web services performance using priority allocation method. pages 201–206. IEEE Computer Society, 2005.

[70] D.J. Kim, D.L. Ferrin, and H.R. Rao. A study of the effect of consumer trust on consumer expectations and satisfaction: The Korean experience. In *5th International Conference on Electronic Commerce (ICEC 2003)*, pages 310–315. ACM, 2003.

[71] SB Kotsiantis, D. Kanellopoulos, and PE Pintelas. Data preprocessing for supervised leaning. *International Journal of Computer Science*, 1(2):111–117, 2006.

[72] BR Kowalski and CF Bender. K-nearest neighbor classification rule (pattern recognition) applied to nuclear magnetic resonance spectral interpretation. *Analytical Chemistry*, 44(8):1405–1411, 1972.

[73] K. Lakshminarayan, S.A. Harp, and T. Samad. Imputation of missing data in industrial databases. *Applied Intelligence*, 11(3):259–275, 1999.

[74] C. Li, G. Peng, K. Gopalan, and T. Chiueh. Performance guarantee for cluster-based internet services. In *9th International Conference on Parallel and Distributed Systems (ICPADS 2003)*, pages 327–332. IEEE, 2003.

[75] Q. Li, B.M. Kim, and S.H. Myaeng. Clustering for probabilistic model estimation for CF. In *14th International Conference on World Wide Web (WWW 2005)*, pages 1104–1105. ACM, 2005.

[76] Z. Liang and W. Shi. Analysis of ratings on trust inference in open environments. *Performance Evaluation*, 65(2):99–128, 2008.

[77] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[78] R.J.A. Little and D.B. Rubin. *Statistical analysis with missing data*. Wiley New York, 2002.

[79] J. Liu, Q. Zhang, B. Li, W. Zhu, and J. Zhang. A unified framework for resource discovery and QoS-aware provider selection in ad hoc networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(1):13–21, 2002.

[80] P. Liu, E. El-Darzi, L. Lei, C. Vasilakis, P. Chountas, and W. Huang. An analysis of missing data treatment methods and their application to health care dataset. In *Advanced Data Mining and Applications*, pages 583–590. Springer, 2005.

[81] Y. Liu, A.H. Ngu, and L.Z. Zeng. QoS computation and policing in dynamic web service selection. In *13th International World Wide Web Conference (WWW 2004)*, pages 66–73. ACM, 2004.

[82] H. Ludwig. Web services QoS: external SLAs and internal policies or: how do we deliver what we promise? In *4th International Conference on Web Information Systems Engineering Workshops (WISEW 2003)*, pages 115–120. IEEE, 2003.

[83] H. Ludwig, A. Dan, and R. Kearney. Crona: an architecture and library for creation and monitoring of WS-agreents. In *2nd International Conference on Service Oriented Computing (ICSOC 2004)*, pages 65–74. ACM, 2004.

[84] H. Ludwig, A. Keller, A. Dan, R. King, and R. Franck. A service level agreement language for dynamic electronic services. *Electronic Commerce Research*, 3(1):43–59, 2003.

[85] Z. Malik and A. Bouguettaya. Rater credibility assessment in web services interactions. *World Wide Web*, 12(1):3–25, 2009.

[86] A. Mani and A. Nagarajan. Understanding quality of service for web services. *IBM developerWorks*, 2002.

[87] O. Martín-Díaz, A. Ruiz-Cortés, A. Durán, D. Benavides, and M. Toro. Automating the procurement of web services. In *1st International Conference on Service-Oriented Computing (ICSOC 2003)*, pages 91–103. Springer, 2003.

[88] O. Martín-Díaz, A. Ruiz-Cortés, A. Durán, and C. Müller. An approach to temporal-aware procurement of web services. In *3rd International Conference on Service-Oriented Computing (ICSOC 2005)*, pages 170–184. Springer, 2005.

[89] E.M. Maximilien and M.P. Singh. A framework and ontology for dynamic web services selection. *Internet Computing, IEEE*, 8(5):84–93, 2004.

[90] E.M. Maximilien and M.P. Singh. Toward autonomic web services trust and selection. In *2nd International Conference on Service Oriented Computing (ICSOC 2004)*, pages 212–221. ACM, 2004.

[91] E.M. Maximilien and M.P. Singh. Agent-based trust model involving multiple qualities. In *4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005)*, pages 519–526. ACM, 2005.

[92] J.S. Milton and J.C. Arnold. *Introduction to probability and statistics: principles and applications for engineering and the computing sciences*. McGraw-Hill, Inc. New York, NY, USA, 2002.

[93] G. Molenberghs, H. Thijs, I. Jansen, C. Beunckens, M.G. Kenward, C. Mallinckrodt, and R.J. Carroll. Analyzing incomplete longitudinal clinical trial data. *Biostatistics*, 5(3):445, 2004.

[94] C. Molina-Jimenez, S. Shrivastava, J. Crowcroft, and P. Gevros. On the monitoring of contractual service level agreements. In *1st International Workshop on Electronic Contracting*, pages 1–8. IEEE, 2004.

[95] G. Morgan, S. Parkin, C. Molina-Jimenez, and J. Skene. Monitoring middleware for service level agreements in heterogeneous environments. In *Challenges of Expanding Internet: E-Commerce, E-Business, and E-Government*, pages 79–93. Springer, 2005.

[96] L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation. In *35th Annual Hawaii International Conference on System Sciences (HICSS 2002)*, pages 2431–2439. IEEE Computer Society, 2002.

[97] I. Myrtveit, E. Stensrud, and U. Olsson. Assessing the benefits of imputing ERP projects with missing data. In *7th International Software Metrics Symposium*, pages 78–84. IEEE Computer Society, 2001.

[98] I. Myrtveit, E. Stensrud, and U.H. Olsson. Analyzing data sets with missing data: An empirical evaluation of imputation methods and likelihood-based methods. In *IEEE Transactions on Software Engineering*, pages 999–1013. IEEE Computer Society, 2001.

[99] D.V. Nguyen, N. Wang, and R.J. Carroll. Evaluation of missing value estimation for microarray data. *Journal of Data Science*, 2(4):347–370, 2004.

[100] U. Oasis. Introduction to UDDI: Important features and functional concepts. *Technical Report, Publisher: Organisation for the Advancement of Structured Information Standards*, October 2004.

[101] A.P. Oodan. *Telecommunications quality of service management: from legacy to emerging services*. IET, 2003.

[102] M.P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann. Service-oriented computing: State of the art and research challenges. *Computer*, 40(11):38–45, 2007.

[103] A. Parasuraman, V.A. Zeithaml, and L.L. Berry. Reassessment of expectations as a comparison standard in measuring service quality: implications for further research. *The Journal of Marketing*, 58(1):111–124, 1994.

[104] R.G. Pearson, W. Thuiller, M.B. Araújo, E. Martinez-Meyer, L. Brotons, C. McClean, L. Miles, P. Segurado, T.P. Dawson10, and D.C. Lees11. Model-based uncertainty in species range prediction. *Journal of Biogeography*, 33:1704–1711, 2006.

[105] D. Pyle. *Data preparation for data mining*. Morgan Kaufmann, 1999.

[106] J.R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[107] J.R. Quinlan. Combining instance-based and model-based learning. In *10th International Conference on Machine Learning (ICML 1993)*, pages 236–243. Morgan Kaufmann, 1993.

[108] A. Ragel and B. Cremilleux. MVC–a preprocessing method to deal with missing values. *Knowledge-Based Systems*, 12(5-6):285–291, 1999.

[109] E. Rahm and H.H. Do. Data cleaning: Problems and current approaches. In *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, volume 16, pages 3–14. Citeseer, 2000.

[110] S. Ran. A model for web services discovery with QoS. *ACM SIGecom Exchanges*, 4(1):1–10, 2003.

[111] S. Reece, S. Roberts, A. Rogers, and N.R. Jennings. A multi-dimensional trust model for heterogeneous contract observations. In *22nd Conference on Association of Artificial Intelligence (AAAI 2007)*, volume 22, pages 128–135. AAAI Press, 2007.

[112] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *the 1994 ACM Conference on Computer Supported Cooperative Work*, pages 175–186. ACM, 1994.

[113] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation systems. *Communications of the ACM*, 43(12):45–48, 2000.

[114] E.S. Ristad. A natural law of succession. *Arxiv preprint cmp-lg/9508012*, 1995.

[115] D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, and D. Fensel. Web service modeling ontology. *Applied Ontology*, 1(1):77–106, 2005.

[116] W. Rong, K. Liu, and L. Liang. Personalized web service ranking via user group combining association rule. In *7th International Conference on Web Services (ICWS 2009)*, pages 445–452. IEEE Computer Society, 2009.

[117] J. Sabater and C. Sierra. REGRET: reputation in gregarious societies. In *5th International Conference on Autonomous Agents*, pages 194–195. ACM, 2001.

[118] C. Saha and M.P. Jones. Bias in the last observation carried forward method under informative dropout. *Journal of Statistical Planning and Inference*, 139(2):246–255, 2009.

[119] A. Sahai, A. Durante, and V. Machiraju. Towards automated SLA management for Web services. *Hewlett-Packard Research Report HPL-2001-310 (R. 1)*.

[120] A. Sahai, J. Ouyang, V. Machiraju, and K. Wurster. Specifying and guaranteeing quality of service for web services through real time measurement and adaptive control. *E-Services Software Research Department, HP Laboratories, Palo-Alto, E-Service Management Project*, 2001.

[121] I.G. Sande. Imputation in surveys: coping with reality. *The American Statistician*, 36(3):145–152, 1982.

[122] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *10th International Conference on World Wide Web (WWW 2001)*, pages 285–295. ACM, 2001.

[123] A.I. Schein, A. Popescul, L.H. Ungar, and D.M. Pennock. Methods and metrics for cold-start recommendations. In *25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260. ACM, 2002.

[124] M.A. Serhani, R. Dssouli, A. Hafid, and H. Sahraoui. A QoS broker based architecture for efficient Web services selection. In *3rd International Conference on Web Services (ICWS 2005)*, pages 113–120. IEEE Computer Society, 2005.

[125] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei. Personalized QoS prediction forweb services via collaborative filtering. In *5th International Conference on Web Services (ICWS 2007)*, pages 439–446. IEEE Computer Society, 2007.

[126] W. Sherchan, S. Krishnaswamy, and S.W. Loke. Relevant past performance for selecting web services. In *5th International Conference on Quality Software (QSIC 2005)*, pages 439–445. IEEE Computer Society, 2005.

[127] G. Shercliff, J. Shao, W.A. Gray, and N.J. Fiddian. A multiple quality-space mapping approach to QoS assessment. In *6th International Conference on Computer and Information Technology (CIT 2006)*, pages 26–26. IEEE Computer Society, 2006.

[128] J. Skene, A. Skene, J. Crampton, and W. Emmerich. The monitorability of service-level agreements for application-service provision. In *6th International Workshop on Software and Performance*, pages 3–14. ACM, 2007.

[129] C. Song and Q. Zhang. Sliding-window algorithm for asynchronous cooperative sensing in wireless cognitive networks. In *International Conference on Communications (ICC 2008)*, pages 3432–3436. IEEE, 2008.

[130] P. Stockreisser, J. Shao, W. Gray, and N. Fiddian. Supporting QoS monitoring in virtual organisations. In *4th International Conference on Service-Oriented Computing (ICSOC 2006)*, pages 447–452. Springer, 2006.

[131] D.L. Streiner. Missing data and the trouble with LOCF. *Evidence Based Mental Health*, 11(1):3, 2008.

[132] K. Strike, K. El Emam, and N. Madhavji. Software cost estimation with incomplete data. *IEEE Transactions on Software Engineering*, pages 890–908, 2001.

[133] O. Svenson. Process descriptions of decision making. *Organizational behavior and human performance*, 23(1):86–112, 1979.

[134] L. Taher, R. Basha, and H. El Khatib. Establishing Association between QoS Properties in Service Oriented Architecture. In *1st International Conference on Next Generation Web Services Practices (NWeSP 2005)*, page 6. IEEE Computer Society, 2005.

[135] W.T.L. Teacy, J. Patel, N.R. Jennings, and M. Luck. Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198, 2006.

[136] M. Tian, A. Gramm, H. Ritter, and J. Schiller. Efficient selection and monitoring of QoS-aware web services with the WS-QoS framework. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI 2004)*, pages 152–158. IEEE Computer Society, 2004.

[137] T. Townsend-Weber and D. Kibler. Instance-based prediction of continuous values. In *Working Notes of the AAAI94 Workshop on Case-Based Reasoning (AAAI 1994)*, pages 30–35. AAAI Press, 1994.

[138] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R.B. Altman. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17(6):520, 2001.

[139] S. Tseng, K. Wang, and C. Lee. A pre-processing method to deal with missing values by integrating clustering and regression techniques. *Applied Artificial Intelligence*, 17(5):535–544, 2003.

[140] B. Twala, M. Cartwright, and M. Shepperd. Comparison of various methods for handling incomplete data in software engineering databases. In *International Symposium on Empirical Software Engineering (ISESE 2005)*, page 10. IEEE Computer Society, 2005.

[141] L.H. Vu and K. Aberer. Towards Probabilistic Estimation of Quality of Online Services. In *7th International Conference on Web Services (ICWS 2009)*, pages 99–106. IEEE Computer Society, 2009.

[142] L.H. Vu, M. Hauswirth, and K. Aberer. QoS-based service selection and ranking with trust and reputation management. In *Cooperative Information System Conference (CoopIS 2005)*, pages 466–483. Springer, 2005.

[143] X. Wang, T. Vitvar, M. Kerrigan, and I. Toma. A QoS-aware selection model for semantic web services. In *4th International Conference on Service-Oriented Computing (ICSOC 2006)*, pages 390–401. Springer, 2006.

[144] Y. Wang and I.H. Witten. Inducing model trees for continuous classes. In *9th European Conference on Machine Learning Poster Papers*, pages 128–137, 1997.

[145] A. Whitby, A. Jøsang, and J. Indulska. Filtering out unfair ratings in bayesian reputation systems. In *3rd International Joint Conference on Autonomous Agenst and Multi Agent Systems (AAMAS 2004)*, pages 48–64. ACM, 2004.

[146] D.R. Wilson and T.R. Martinez. Reduction techniques for exemplar-based learning algorithms. *Machine Learning*, 38(3):257–286, 2000.

[147] E. Wohlstadter, S. Tai, T. Mikalsen, I. Rouvellou, and P. Devanbu. Glueqos: Middleware to sweeten quality-of-service policy interactions. IEEE Computer Society, 2004.

[148] R. Wolski, N.T. Spring, and J. Hayes. The network weather service: a distributed resource performance forecasting service for metacomputing. *Future Generation Computer Systems*, 15(5-6):757–768, 1999.

[149] G. Wu, J. Wei, X. Qiao, and L. Li. A bayesian network based QoS assessment model for web services. In *4th International Conference on Services Computing (SCC 2007)*, pages 498–505. IEEE Computer Society, 2007.

[150] L. Xiong and L. Liu. Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):843–857, 2004.

[151] Z. Xu, P. Martin, W. Powley, and F. Zulkernine. Reputation-enhanced qos-based web services discovery. In *5th International Conference on Web Services (ICWS 2007)*, pages 249–256. IEEE Computer Society, 2007.

[152] D. Yang, Z. Wu, B. Yan, D. Qian, and Z. Luan. Agent-based MADM approach to the dynamic web service selection. In *2nd Asia-Pacific Service Computing Conference (APSCC 2007)*, pages 260–265. IEEE Computer Society, 2007.

[153] B. Yu and M.P. Singh. An evidential model of distributed reputation management. In *1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*, pages 294–301. ACM, 2002.

[154] B. Yu and M.P. Singh. Detecting deception in reputation management. In *2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2003)*, pages 73–80. ACM, 2003.

[155] B. Yu and M.P. Singh. Searching social networks. In *2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2003)*, pages 65–72. ACM, 2003.

[156] Y. Yu, I. Cheng, and A. Basu. Optimal adaptive bandwidth monitoring for QoS based retrieval. *IEEE Transactions on Multimedia*, 5(3):466–472, 2003.

[157] G. Zacharia and P. Maes. Trust management through reputation mechanisms. *Applied Artificial Intelligence*, 14(9):881–907, 2000.

[158] V.A. Zeithaml, A. Parasuraman, and L.L. Berry. *Delivering quality service: Balancing customer perceptions and expectations*. Free Pr, 1990.

[159] C. Zhang, Q. Yang, and B. Liu. Guest Editors' Introduction: Special Section on Intelligent Data Preparation. *IEEE Transactions on Knowledge and Data Engineering*, 17(9):1163–1165, 2005.

[160] J. Zhang and R. Cohen. A personalized approach to address unfair ratings in multiagent reputation systems. In *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006) Workshop on Trust in Agent Societies*, 2006.

[161] S. Zhang, C. Zhang, and Q. Yang. Data preparation for data mining. *Applied Artificial Intelligence*, 17(5):375–381, 2003.

[162] Z. Zheng, H. Ma, M.R. Lyu, and I. King. Wsrec: A collaborative filtering based web service recommender system. In *7th International Conference on Web Services (ICWS 2009)*, pages 437–444. IEEE Computer Society, 2009.