2013 IEEE Ninth World Congress on Services

An End-To-End QoS Mapping Approach for Cloud Service Selection

Raed Karim, Chen Ding
Department of Computer Science
Ryerson University, Toronto, Canada
r2karim@ryerson.ca, cding@scs.ryerson.ca

Ali Miri
Department of Computer Science
Ryerson University, Toronto, Canada
ali.miri@ryerson.ca

***Abstract*— In order to select and rank the best services in a cloud computing environment, the end-to-end quality of service (QoS) values of cloud services have to be computed. For a new SaaS provider, the deployment of its software application in the cloud is a challenging job. It has to find a hosting service (IaaS) that hosts its service. The primary goal of the SaaS provider is to make its service at the top of the ranked list of cloud services returned to end users through satisfying their QoS requirements. In this paper, we propose a mechanism to map the users' QoS requirements of cloud services to the right QoS specifications of SaaS then map them to best IaaS service that offers the optimal QoS guarantees. Then together SaaS and IaaS services can provide the best service offer to end users. As a result of the mapping, the end-to-end QoS values can be calculated. We propose a set of rules to perform the mapping process. We hierarchically model the QoS specifications of cloud services using the Analytic Hierarchy Process (AHP) method. The AHP based model helps to facilitate the mapping process across the cloud layers, and to rank the candidate cloud services for end users. We use a case study to illustrate and validate our solution approach.**

Keywords- SaaS; IaaS; QoS mapping; QoS-based web Service Selection; Cloud Computing; Analytic Hierarchy Process

I. INTRODUCTION

Cloud computing represents a network of datacenters that constitutes a large number of computers working collaboratively to perform and provide users with large collections of applications and business functions [1]. The important characteristic of cloud computing is the ability to provision different types of software as on demand, affordable and elastic services to consumers [2]. Software providers can have access to advanced technology without incurring high capital expenses on hardware acquisition. And they can reach a large number of consumers around the world. For cloud users, they now can access a large variety of software applications available through the Internet on pay-per-use basis.

Cloud providers with different service models (IaaS, PaaS, SaaS, or other cloud services) have to make sure they deliver high quality services. The goal is to meet users' expectations and to compete with their peers who provide functionally equivalent services. Service quality constitutes different computing aspects such as service performance (availability, reliability and response time), business and economic (price, usability and reputation), trust and security [3]. In web service selection, QoS is the main criteria that a service selection system uses to select and rank the best service candidates for end users. we consider the Cloud Service Measurement Index (CSMI) that includes a set of business-relevant key performance attributes [4].

In this paper, we address the problem of QoS based cloud services selection and ranking. Other works have addressed the problem including [5][6][7]. However, they assume that users' QoS requirements (submitted to the selection systems) are end-to-end. Similar to traditional (non-cloud) selection systems, they just optimize the claimed QoS values to find a ranked services list. The ranking components of these systems consider services that belong to a specific layer (e.g. SaaS or IaaS). Then the system aggregates the QoS values claimed by each service provider to find a service that best optimizes the claimed values. In one hand and real life scenario, users' QoS requirements have to be considered when selecting cloud services. On the other hand, users do not always have the expertise to provide all types of requirements, specifically, the ones with a high technical nature. In our approach we consider mapping the QoS requirements (submitted by end users) across the necessary cloud layers. We consider the SaaS's perspective to help them map the users' requirements to their QoS guarantees then perform the mapping again to the most appropriate IaaS service available in the Internet. Together, SaaS and IaaS, as a combined cloud service can offer the best service offer to end users. Then we use these values along with the generated QoS weights from the AHP [8] method to rank the set of the cloud services (combined services) for end users. We hierarchically model the QoS specifications of cloud services using AHP method to facilitate the mapping process across different cloud layers (i.e. IaaS and SaaS layers) besides end users layer.

The main contributions of this paper are: 1) We propose a novel approach for cloud service selection and ranking. It is based on mapping the user's QoS requirements to SaaS layer then we allow SaaS to map these requirements to select the best IaaS service. 2) For this purpose, we model the complex QoS specifications of multiple cloud layers including QoS requirements and their measurement metrics.

The rest of the paper is organized as follows. Section II reviews the related work on QoS based cloud web service selection. Section III presents our view to the QoS specifications modeling using AHP method. Section IV presents our mechanism of QoS mapping and computation in the cloud. Section V introduces our service selection and ranking algorithm. Section VI presents a detailed case study to illustrate our approach. Finally, section VII concludes the paper and presents the future work.

II. RELATED WORK

Various approaches were proposed for QoS based service selection. In this section we consider three different areas of research that are more related to our research work. They are: QoS based service selection, QoS based cloud service selection and QoS mapping approaches.

Considering the web service selection systems, in [9], a QoS-context aware model is proposed to overcome the difficulty of modeling some QoS attributes. In the model, QoS context factors are extracted by analyzing their correlation with QoS attributes. Then a web service selection model which is based on prediction and user requirements is proposed. Three main steps are defined in the process: context information generation, QoS prediction and service selection based on QoS prediction. In [10], the authors introduce an enhancement to the PROMETHEE algorithm (one of the Multi-Criteria Decision Making methods) for QoS based web service selection. The main focus of the paper is on handling the user' s QoS request in the selection and ranking process, and considering the interdependency relationships between the QoS attributes. In [11], for the QoS specifications that contain only a linear constraint, Mixed Integer Programming(MIP) is implemented whereas Constraint Programming(CP) is used for non-linear constraints. Based on the experiments conducted in this paper, the MIP approach outperformed the CP one when considering linear constraints. The process of matchmaking was achieved using MIP or a CP engine depending on the type of constraints (linear or nonlinear respectively). In [12], an adaptive selection framework is proposed in which the users define a utility function that describes a quality of service attribute. Then a proposed learning policy is used to explore and exploit interesting services. This is to maximize the user's utility function according to her requirements. After calculating the utility function for each QoS, the user attempts to interact with a provider that can maximize her utility function.

Considering the cloud computing environments, some significant research works have focused on the cloud service selection in the recent years. In [6], a cloud service selection approach based on user's constraints on QoS is proposed. For modeling the QoS, the paper considers the service management index (SMI) attributes in which the QoS attributes are classified into seven major groups. A framework is developed that handles the QoS management, monitoring and ranking the services. The AHP method is used for ranking the candidate services. In the ranking process, the user requirements on both SaaS and IaaS service are presented. Also, QoS calculation models are presented to calculate the QoS values for the ranking process. In [7], the authors propose an optimization model of service selection for multi-tenant SaaS. SaaS developers compose appropriate services based on different QoS constraints of multiple users. It also considers achieving SaaS optimizing goal (less price and high performance). This is performed by modeling the problem as constraint optimization problem (COP). In [8], a brokerage based cloud service selection framework is proposed that consists of three parties: a cloud provider, a cloud user and a broker. The latter extracts the QoS attributes from the providers and QoS requirements from users. Then it indexes the providers according to their similarities. In [13], the authors propose a brokering based semantic cloud service selection system which is based on OWL-S. The system supports the complex service constraints using SWRL rules. It also supports the dynamic matching of services via ontology representation. The selection process is based on the QoS matchmaking that has four matching levels (Exact, Subsume, Partial, Invert and Fail). The services' scores are obtained based on QoS matching types. Then the resulted list of services is ranked based on their scores.

Another trend of cloud service selection is through establishing a cloud marketplace. A designed broker or mediator plays the main role in managing the interaction between service providers and potential consumers. The different roles that a broker plays are monitoring service data, prediction, negotiation, selection and ranking services. In the selection part a broker can perform different other activities such as modeling pricing and optimizing the resources allocation in order to select the most appropriate services to end users. There are many research proposals that adopt the marketplace approach for selecting the best service [14] [15] [16].

The approach of mapping quality parameters are used in other domains such as networks, mobile networks and multimedia systems. In [17], the authors propose a high level abstract framework to combine the capabilities of two network architectures; the Integrated Services and Differentiated Services. The end users connect to IntServ networks and the latter interconnects with DiffServ. For this purpose, the paper introduces a formal approach for QoS handling between the two architectures through QoS requirements and mapping mechanisms. In [18], the paper proposes a QoS mapping and adaptation mechanism to map QoS requirements to heterogeneous networks. The two main characteristics of the mapping mechanism are considering multi-users mapping and QoS adaptation over the heterogeneous mobile networks. In [19], A QoS mapping model is proposed to map different service specifications across different network providers. These specifications are the traffic flow, traffic flow and QoS (e.g. delay jitter and packet loss probability). A third part agent classifies the service specifications into multiple classes then a proposed mapping algorithm selects the most appropriate class for application requirements.

From the above description, existing approaches for service selection and QoS mapping are not suitable for addressing the problem of QoS based cloud service selection. We summarize our point of view as follows:

1) With respect to the traditional web service selection system, the selection mechanism applies on a single computing layer that represents the service providers. A software provider possesses underlying computing

components (e.g. development platforms, hardware, servers, etc). So, the QoS guarantees are offered within one layer. However, in cloud environments, a provisioned service is composed of other services from different layers of models. Consequently, end-to-end QoS values should be computed based on these computing layers.

2) In cloud related service selection models we have found that most of the existing proposals do not modify their selection systems to accommodate the new computing architecture of the cloud that independently models the services in multiple layers.

3) The mapping schemes proposed in other domains are difficult to adopt in our problem. There are different reasons such as systems' limitation in terms of representing one computing layer, their focus on QoS extraction and negotiation process and, their lack of presenting a clear road map for mapping QoS attributes.

Our main motivation for proposing our QoS based cloud service selection is to solve the aforementioned shortcomings and disadvantages. In our approach, we capture the complexity of QoS requirements and guarantees in multiple cloud layers at which different types of services are offered. Our proposed approach helps SaaS providers map the users' QoS requirements to their requirements then map the latter to the IaaS layer. The goal is to satisfy the users' QoS requirements by selecting the best IaaS service. Together, the combined services can offer the best cloud service to the users through our selection system.

III. A HIERARCHICAL MODEL FOR QOS SPECIFICATIONS IN THE CLOUD

Based on the above description, computing the end-to-end QoS requirements in a dynamic cloud environment is a challenging task. The QoS computing process can be more complex if we consider many more cloud layers (e.g. PaaS, DBaaS, DaaS, etc). We propose a QoS mapping mechanism that uses a set of rules to map the QoS specifications (QoS requirements and guarantees) across the cloud layers. In order to facilitate the mapping and develop mapping rules, we build a hierarchy to capture the complexity of QoS specifications of the cloud services and their relationships.

A. Analytic Hierarchy Process (AHP)

AHP: is a well-known multi criteria decision making (MCDM) method that models and solves problems of different domains. This type of problems has three main elements (goal, criteria and a solution space that has several alternatives). In our problem domain the goal is to select and rank cloud services, the criteria are the QoS requirements, and alternatives solutions are the offered cloud services. So, AHP is a natural solution approach for our problem. Using AHP, we hierarchically structure the different QoS specifications and their top-down interdependency relationships in multiple cloud layers. This capability highly facilitates mapping the users' requirements to other QoS requirements and guarantees in the cloud environment. In addition, we use AHP to generate priorities of the QoS attributes which represent their degrees of relative importance with respect to the goal.

Fig.1 illustrates our AHP based modeling of the QoS specifications in the cloud environment. The model depicts the top-down dependency relationships between different QoS attributes in three cloud layers (Users, SaaS and IaaS). The model also captures some deeper dependency relationships between the attributes and their metrics. The hierarchy consists of multiple levels that represent the cloud layers. Each level is composed of classes and their subclasses. The classes and subclasses represent the QoS specifications and their metrics. The classes are connected through the levels with dependency relationships. The metrics are used as parameters in the QoS computing process. As shown in the figure, we decompose the problem into its constituent elements through top (goal) level, intermediate levels (QoS attributes and metrics at multiple cloud layers), and bottom (service alternatives) level. The immediate level below the goal has three main classes that refer to the three main QoS groups: Performance-related QoS, Customer-related QoS and System QoS. The following level has two subclasses of the first and second groups: Stability and Accuracy, respectively. The level below has multiple subclasses that represent the QoS attributes under Stability and Accuracy classes. Then, the immediate level represents QoS specifications of SaaS. The following level has SaaS's subclasses that represent QoS metrics to measure QoS guarantees of SaaS services. In this level, IaaS's QoS specification subclasses are also presented. In the level below, QoS metrics that measure the QoS guarantees of IaaS are presented. Our system exploits this important capability of AHP to facilitate the development of a set of mapping rules.

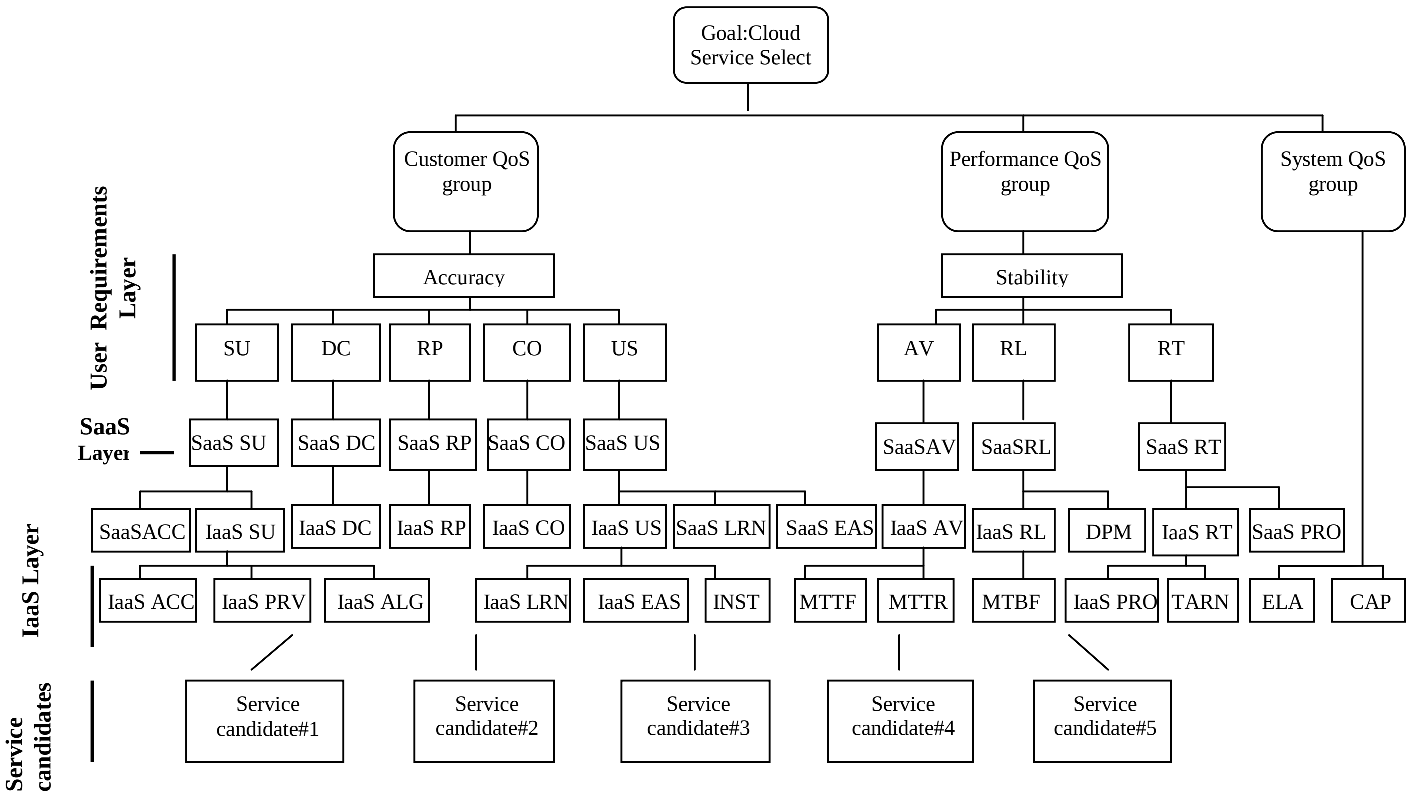
IV. QOS MAPPING RULES AND COMPUTATION

As we explained, the QoS mapping rules are used to map the QoS requirements across the cloud so our system can calculate the QoS values of the offered cloud services. We formulate the required formulas according to each rule to calculate the end-to-end QoS values. In this paper, we present a new way of calculating the end-to-end QoS values. The SaaS's QoS values are first computed using the metrics associated with SaaS services. Then based on the used mapping rule, the IaaS values are calculated by computing the values of the IaaS's QoS metrics. The system aggregates the two values using some aggregation functions. We adopt the aggregation functions used in the sequential service composition system to combine the QoS values [20]. The reason for using the sequential composition is because the SaaS-based QoS value needs to be calculated and combined with the IaaS one after hosting the application in an IaaS service. We define three rules as follows:

Rule #1: the user's QoS requirements are either mapped to SaaS layer then to the IaaS layer or they are mapped to IaaS layer directly. The mapping could further extend to include QoS metrics in both layers. Two sub-rules can be extracted as follows:

Let Q refer to the set of all users' QoS requirements.

Sub-Rule #1: the SaaS provider maps the users'



Acronyms. SU:security,DC:data control,RP:service reputation:CO:service cost,US:service usability,AV:service availability,RL:service reliability,RT: service response time,ACC: sccess control methods,PRV: privacy,ALG:encryption algorithms,LRN:service learnability,EAS: service ease of use, INST: IaaS installability, MTTF: mean time to fail,MTTR:mean time to repair, PRO:request processing time, TRAN: transmission time, DPM: defects per million, MTBF: mean time between fail, ELA: service elasticity,CAP: capacity.

☐ Classes ——— Dependency relations   Figure.1 AHP Model for QoS Specifications of Cloud Services.

requirements to IaaS layer since SaaS has no control on this type of QoS requirements. Some example of this scenario is *AV* and *DC*. Figure. 2 depicts the mapping under this rule. We model the rule as follows:

$QU_i \rightarrow QI_n$ where $QU_i \subseteq Q$ refers to the users' QoS requirements that fit sub-rule #1, QI_n refers to IaaS's QoS specifications of sub-rule #1 in which $n \in Z$.

Sub-Rule #2: the SaaS provider maps the users' requirements to its QoS requirements then to the IaaS' guarantees. The end-to-end QoS values are a combination of all declared or measured values at SaaS and IaaS layers. Some examples are *RL*, *US*, *CO*. Figure. 3 depicts the mapping under this rule. We model the rule as follows:

$QU_j \rightarrow QS_m \rightarrow QI_m$, where $QU_j \subseteq Q$ refers to the users' QoS of sub-rule #2, QS_m and QI_m refer to SaaS's and IaaS's QoS specifications respectively where $m \in Z$.

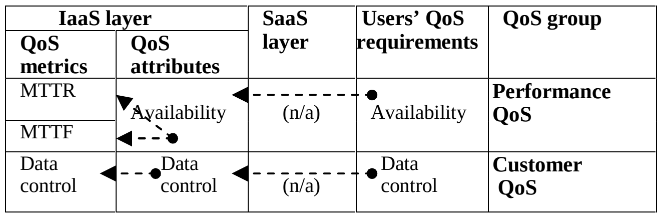


Figure 2. QoS mapping Sub-Rule#1 of Rule#1.

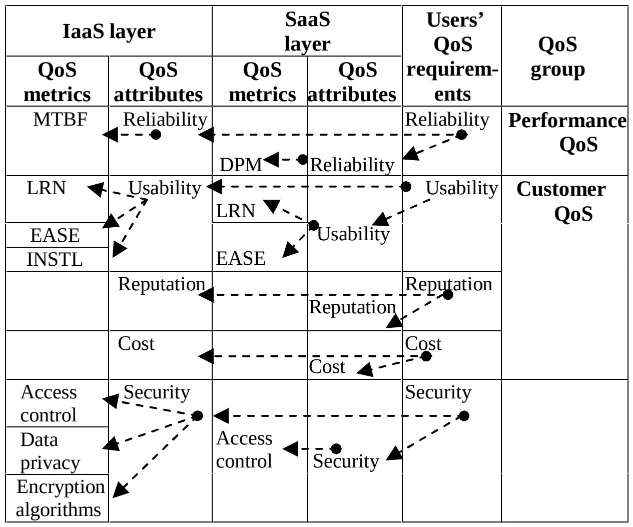


Figure 3. QoS mapping Sub-Rule#2 of Rule#1.

Below we show the computation of rule #1. We provide examples of the selected QoS through this paper and their calculations.

Availability is the degree of the service accessibility by its users. The service availability is computed based on the measured availability of the hosting IaaS service (VMs). It means that availability can only be granted by VMs. So (*AV = IaaS AV*). Formula 1 [21] is used for this purpose. The unit is percentage and the tendency is high.

$$IaaS\ AV = MTTF\ /\ (MTTF + MTTR) \quad (1)$$

where MTTF stands for Mean Time To Fail which refers to the amount of time a service is up before it fails, and MTTR stands for Mean Time To Repair which refers to the time a service needs to return to a working status after failing.

Data control refers to the provider's mechanisms that guarantee a certain level of control for users over their data stored and processed in the cloud. The requested value of data control is computed based on the IaaS value since the hosting services is completely responsible for storing users' data. So (*DC = IaaS DC*). We assume that the standard policies for maintaining the data control specification in the cloud are predefined. The similarity between the claimed policies by IaaS provider and the predefined policies are measured. A service that has the highest similarity will be the best. We use Formula 2 for the calculation. The unit is percentage and the tendency is high.

$$IaaS\ DC = sim\left(\frac{Service\,\Pr ovider\ \text{specification of DC policy}}{\text{Pre}-\text{determined DC policy}}\right) * 100\% \quad (2)$$

The computation of the sub-rule #2 is more complex. This rule requires a combination of multiple QoS values from different clouds (i.e. SaaS and IaaS providers) to compute the end-to-end QoS values. We have mentioned earlier that the aggregation functions of the service composition system can be used for this purpose. Below we show the QoS computation of sub-rule#2.

Reliability: it refers to the service ability to function according to the agreed upon performance requirements in SLA. We use formula 3 to compute the cloud service reliability. The end-to-end value of reliability is a product of IaaS's and SaaS's values.

$$RL = \prod(SaaS\ RL, IaaS\ RL) \quad (3)$$

SaaS'S reliability can be calculated using DPM factor that refers to the number of defects per million attempts of user's requests. The less DPM value the better the software is. The DPM is calculated as shown in formula 4 [22].

$$DPM = \frac{Unsuccessful\ requests}{Total\ request} * 1000,000 \quad (4)$$

DPM is transformed to reliability which is measured using percentage as shown in formula 5.

$$SaaS\ RL = \left(\frac{1000{,}000 - \text{DPM}}{1000{,}000}\right) * 100\% \quad (5)$$

To calculate the IaaS reliability value, the MTBF metric is computed as shown in formula 6 [23]:

$$MTBF = \left(\frac{Total\ \text{time between failure}}{Number\ \text{of failures}}\right) * 100\% \quad (6)$$

MTBF stands for Mean Time Between Failures which refers to the average time between failures. The reliability unit is percentage and tendency is high.

Response Time: it is measured by computing the difference in time between sending a request and receiving a response (a service is made available). The cloud response time is measured in formula 7. The end-to-end value of response time is a summation of IaaS's and SaaS's values.

$$RT = (SaaS\ \text{RT} + IaaS\ \text{RT}) \quad (7)$$

The SaaS response time is computed by measuring the processing time for completing a request. It is measured without considering communication time with the hosting server. Formula 8 is used for the calculation.

$$SaaS\ RT = SaaS\ PROC \quad (8)$$

The IaaS's response time is computed by combining the processing time and transmission time. The processing time includes time to process functions, waiting time and time for service composition. The transmission time is the total time for communicating with the hosting server. Formula 9 is used for the calculation.

$$IaaS\ RT = IaaS\ PROC + TRANS \quad (9)$$

The measurement unit is seconds and tendency is low.

Usability: it is the cloud service characteristic of being easy to use, learn and install. We use formula 10 to measure the cloud usability. The end-to-end value of usability is a product of IaaS's and SaaS's values.

$$US = \prod(SaaS\ \text{US}\ , \text{IaaS US}) \quad (10)$$

To measure the usability of SaaS and IaaS services, we consider the users' feedback to rate the services based on the service's ease of use, learnability, and its installability (for IaaS only). The users are given a range of [0-10] where 0 means very poor usability and 10 means excellent usability. The measurement unit is percentage and tendency is high.

Reputation: it is the level of a service's overall acceptance by its users. Cloud service reputation is measured in formula 11. The end-to-end value of reputation is a product of IaaS's and SaaS's values.

$$RP = \prod(SaaS\ \text{RP} + IaaS\ \text{RP}) \quad (11)$$

We consider the users' feedback to rate the services based on their overall performance, prices and usability. The users are given a range of [0-10] where 0 means generally very poor service and 10 means generally excellent service. The measurement unit is percentage and tendency is high.

Security: it is the service characteristic of being secure in terms of applications, hardware components and users' data. We use formula 12 for measuring the cloud security. The end-to-end value of security is a product of IaaS's and SaaS's values.

$$SU = \prod(SaaS\ \text{SU} + IaaS\ \text{SU}) \quad (12)$$

To compute the SaaS security we assume the access control mechanisms (e.g. authorization, authentication) are predefined within a cloud environment. Then we measure the similarity between the access control mechanisms offered by the SaaS provider and the predefined ones. We use formula 13 for this purpose.

$$SaaS\ SU = sim\left(\frac{SaaS\ ACC\ mechanisms}{\Pr e\text{-}defined\ ACC\ mechanisms}\right) \quad (13)$$

IaaS's security is measured considering three factors (i.e. access control, data privacy and encryption algorithms). We use formula 14 for this purpose:

$$IaaS\ SU = sim\left(\frac{IaaS\ \text{ACC, PRIV}\ policies\ \&\ ENCR}{\text{Pr}\,e\text{-}defined\ \text{ACC, PRIV}\ policies\ \&\ ENCR}\right) \tag{14}$$

The measurement unit is percentage and tendency is high.

Cost: it refers to the cost of accessing and using a service that a user should pay. It is measured in formula 15. The end-to-end value of usability is a product of IaaS's and SaaS's values.

$$CO = (SaaS\ \text{CO} + IaaS\ \text{CO}) \tag{15}$$

SaaS cost includes accessing and using cloud applications. IaaS cost refers to the cost of using the hosting service instance which includes Memory, CPU time, data storage, an OS and network bandwidth. The measurement unit is US (dollar) and tendency is low.

Rule #2: some particular QoS requirements are mapped to a group of other requirements representing a defined QoS group in the hierarchical model. In order to compute the values of this type of requirements, the values of the mapped requirements are aggregated. Some examples are *Stability* and *Accuracy*. The former refers to the service ability to remain unchanged in terms of its performance. The latter refers to the degree that service functionality is close to what stated in the service level agreement (SLA). We model the rule as follows. Let QU_{pk} and QU_{ck} be the users' requirements of the performance and customer-related QoS requirements respectively; QI_{pn} and QI_{cn} be IaaS specifications in sub-rule #1, QS_{pm} and QS_{cm} be the SaaS specifications in sub-rule #2, and QI_{pm} and QI_{cm} be the IaaS specifications in rule #2 .

$$\mathbf{QU_{pk} \rightarrow QI_{pn} + (QS_{pm} \rightarrow QI_{pm})}$$
$$\mathbf{QU_{ck} \rightarrow QI_{cn} + (QS_{cm} \rightarrow QI_{cm})}$$

To calculate the two requirements, we specify the required parameters according to their definitions. So, stability measures the service performance and accuracy measures the service functionality and customer related requirements. We use formulas 16 &17 for the calculation. The unit of both requirements is percentage and tendency is high.

$$Stability = \text{avg}(\text{AV} + \text{RL} + \text{RT}) \tag{16}$$

$$Accuracy = \text{avg}(\text{DC} + \text{US} + \text{RP} + \text{CO} + \text{SU}) \tag{17}$$

Rule #3: some SMI attributes have special situations such as capacity (*CAP*) and elasticity (*ELAS*). These attributes are hidden from end users, and only requested by SaaS providers. These attributes are too technical and they are controlled and guaranteed by system providers (i.e. IaaS providers). However, end users are more concerned about the upper level attributes as they are common and known concepts (e.g. availability, usability, response time, etc). A common scenario for using these attributes is when a SaaS provider requires certain values for capacity to accommodate its users' workloads (meeting their QoS requirements). For example, during a peak season SaaS workload jumps to the highest point. From the end user's point of view, the SaaS provider is responsible for maintaining the level of service performance according to SLA such as web site availability of 99.5% of the time. In order to maintain this level of availability, the SaaS provider maps this requirement to IaaS layer so the hosting provider scales up the service capacity. This could be in terms of increasing the number of VMs or changing the configuration of the machine instances. The ability of cloud elasticity has a great impact on services and their economic aspect [24]. The mapping extends to the lower level metrics (computing components) such as CPU, memory, storage capacity and network bandwidth. We model the rule as follows.

Minimize $T = (t_1 – t_0)$, t_0 is initial time, t_1 is scale time end.
Subject to $C\,t_1\,(n) <= C`\,t_0\,(n) \rightarrow Q_i \equiv U_i$.
Variables n = {cpu, memory, bandwidth, data storage}, Ct_1 is the capacity at time t_1, $C`t_0$ is the actual physical capacity at time t_0. $Q_i \in \{QI_{pn}, QI_{pm}\}$, $U_i \in QU_{pk}$.

V. CLOUD SERVICE SELECTION

After computing the end-to-end QoS values, the candidate cloud services are selected and ranked for end users. Beside its ability to hierarchically model QoS specifications of cloud services, AHP can generate the weights required for computing the service selection. We apply the AHP ranking algorithm [5] on the upper three levels of Fig.1 including goal, QoS groups and QoS attributes. We want to generate weights for QoS attributes for which users submit their requirements.

A. AHP based Ranking Algorithm

In Section III, we specify the goal, criteria and solution alternatives. In the following steps, we explain the services ranking algorithm:

Step 1: we decompose our problem to its constituent elements. That is a hierarchy of multiple levels and multiple classes that represent QoS specifications.

Step 2: we construct a set of pair wise comparison matrices. Each class in upper level is used to compare its subclasses in the level below with respect to it. To perform the comparison, a scale of absolute numbers is used. The scale represents the importance of one attribute over other attributes with respect to their parent attribute. The range of these numbers is from 1 to 9 where 1 means two attributes are equally important and 9 means one attribute is extremely more important than the other.

Step 3: we repeat Step 2 for each class (including subclasses that can also be classes with subclasses in the levels below).

Step 4: the output of the matrices is local weights of each class and subclass (QoS attributes). The sum of the local weights is usually equal to 1. We need to obtain global weights for the attributes. Starting from the top level in which classes have their global weights naturally, we multiply the weight of each class in the level below by the weight of its parent class. We repeat the process for each class until we reach the lowest level. The output is global weights of all leaf attributes.

Step 5: to compute the service ranking, for each service candidate, we multiple the calculated end-to-end QoS values with the obtained weights from the previous step. Then we sum up the results (using weighted sum method). The output is services' scores. We rank the services using their scores and return the services' ranked list to the end users. In a previous work [10], we propose to enhance the ranking result by including the user's request of the QoS in the ranking computation process.

TABLE I. USER REQUESTS.

User' requested QoS	
QoS attributes	QoS values
AV	>= 99%
RL	>=98%
RT	<3sec
DC	>=95%
US	>=95%
RP	>=90%
CO	<$1.5
SU	>=97%

TABLE II. QOS VALUES OF SAAS AND IAAS SERVICES.

QoS attributes		AV	RL	RT	DC	US	RP	CO	SU
SaaS QoS values		N/A	DPM (integer)	PROC (sec)	N/A	LRN, EASE (%, %)	RP (%)	CO (US$/hr)	ACCES (%)
IaaS QoS values		MTTR,MTTF (hr, hr)	MTBF (%)	PROC,TRANS (sec, sec)	DC (%)	LRN, EASE, INSTL (%, %, %)	RP (%)	CO (US$/hr)	ACCES, ALG, PRIV (%, %, %)
Cloud services CSs	VM1, S1	-	30	2	-	99, 98	95	1.5	95
		2, 10000	98	1, 1	95	99, 98, 99	99	0.4	99,75,98
	VM2, S1	-	40	2.5	-	97, 97	98	0.7	75
		1, 15000	99	1.5, 1.5	96	99, 99.5, 99	95	0.2	99, 99, 99
	VM3, S1	-	33	1.5	-	99	93	0.8	85
		1.5,12000	98.7	2, 2	93	99, 99, 98	93	0.3	99, 99, 99
	VM4, S1	-	31	1.3	-	97. 97	90	0.9	90
		1, 12000	98.5	1.5, 1.5	94	98, 98, 98	95	0.2	98, 98, 98
	VM5, S1	-	24	1.1	-	98	92	0.6	92
		1.5, 13000	98	1.3, 1	95	99, 98, 98	90	0.1	97, 97, 97

VI. CASE STUDY

The purpose of the case study is to illustrate the process of selecting and ranking the cloud services by using the algorithm of the Section V. The delivered cloud service is a combination of a SaaS services and one of the available IaaS services. Our case study includes different entities: cloud end user, a SaaS and IaaS VMs. The type of services that we assume in this example is as follows: one SaaS service (S1) which is a newly developed software that joins the cloud and IaaS services that provision 4 virtual machines: VMs ={VM1, VM2, VM3, VM4}. The combinations of SaaS and IaaS form the delivered cloud services (CSs). The SaaS provider wants to select the best IaaS service among a list of functionally matching VMs. We assume that a functional matchmaking process has already been performed. Based on past experiences of similar SaaS services and the types of users' QoS requirements, SaaS is able to map these typical requirements to its QoS specifications and to those of IaaS services.

Table. I, II show an example of the input data to our cloud service selection approach. Table.I contains a user's requirements of the QoS that she submitted earlier. Table II contains the SaaS's QoS values and values of multiple VMs that are claimed by their providers or measured by a third party. We assign arbitrary QoS values to the services for illustration purpose. The end-to-end QoS values are calculated based on the initial values of Table. II and using the formulas in Section IV. The obtained values have to be normalized under one scale system so they can be used in the service ranking phase. The normalization can be done by assigning a value of 1 to the offer with a highest value in a QoS column that spans all candidate services. Then each value in the column is divided by the highest value. We repeat the same process for all QoS column. The other element that we need for the ranking process is weights for the end-to-end QoS requirements.

As explained in the previous section, three pair wise comparison matrices are required for our example (Figure.I). We show two matrices in Table III and Table IV.

TABLE III. PAIRWISE COMPARISONS OF QOS ATTRIBUTES W.R.T. CUSTOMER QOS.

QoS attributes	DC	US	RP	CO	SU	QoS weights
DC	1	1	3	3	1	0.283
US	1	1	4	2	1	0.281
RP	1/3	1/4	1	2	1	0.136
CO	1/3	1/4	1/2	1	1	0.104
SU	1	1	1	1	1	0.195

TABLE IV. PAIRWISE COMPARISONS OF QOS ATTRIBUTES W.R.T. PERFORMANCE QOS.

QoS attributes	AV	RT	RL	QoSLocal weights
AV	1	4	1	0.458
RT	1/4	1	1/3	0.126
RL	1	3	1	0.416

The pair wise comparisons of the two matrices are performed with respect to the Customer QoS and Performance QoS respectively. The last column of each matrix shows the QoS weights. The pair wise comparisons process is applied similarly on the two groups with respect to the goal. Then we obtain the global weights of all QoS attributes as explained in the previous sections. Table V shows the computed QoS weights. Finally, for each candidate service, including the added users' requirements

TABLE V. AHP BASED QOS WEIGHTS.

QoS attributes	AV	RL	RT	DC	US	RP	CO	SU
Global weights	0.259	0.229	0.069	0.127	0.126	0.061	0.047	0.088

TABLE VI. CLOUD SERVICES SCORES AND RANKING.

Cloud services (CS)	**Scores**	**CS ranks**
CS2 (VM2, S1)	0.860	1st
CS3 (VM3, S1)	0.847	2nd
CS5 (VM5, S1)	0.840	3rd
CS4 (VM4, S1)	0.835	4th
User's requirements	0.826	
CS1 (VM1, S1)	0.820	5th

we multiply the computed end-to-end QoS values with their weights and add them to get a ranking score. The ranked list (Table VI) is returned to the end users. Obviously, the best service is the one that has the highest score.

VII. CONCLUSIONS

This paper addresses an important problem of handling the QoS requirements of web services in the cloud environment. When a new software service is deployed in the cloud, its goal is not only to satisfy the user's functional requirements but also her QoS requirements. Consequently, the software (SaaS) is selected among other services. The main challenge for SaaS providers is to select the best hosting service (e.g. IaaS provider) in the cloud so both services as a combined cloud service satisfy the users' requested values of the QoS. To achieve this task, SaaS provider has to map their users' requirements to its own QoS specifications then to the IaaS layer. Mapping the QoS requirements helps compute the QoS values. In this paper, we develop a set of mapping rules and present a new way of computing the end-to-end QoS values in the cloud environment. AHP method is used to hierarchically model the QoS specifications, and to generate the QoS weights for the selection and ranking process. In our future steps, we intend to build a brokerage based framework for selecting and ranking services in cloud marketplaces, in which mapping the QoS across the cloud layer is the major component. In this context, we will expand the mapping process to include more layers and more QoS types. We also plan to conduct a performance evaluation experiments based on real QoS dataset of cloud services.

REFERENCES

[1] A. Kalapatapu, M. Sarkar, " Cloud Computing: An Overview", in Cloud Computing: Methodology, Systems, and Applications, L. Wang, R. Ranjan, J. Chen, and B. Benatallah, ISBN: 9781439856413, CRC Press, Boca Raton, FL,USA, October 2011, pp. 3-25.

[2] Y. Wei, M. Blake, " Service-Oriented Computing and Cloud Computing", IEEE Internet Computing, vol. 14, Issue 6, 2010, pp.72-75.

[3] K. Kritikos, D. Plexousakis, "Requirements for QoS-based Web Service Description and Discovery", IEEE Transactionnns on Services Computing, Vol. 2, 2009, pp. 320-337.

[4] The Cloud Sservice Measurement Initiative Consortium, " Service Measurement Index" (SMI), Carnegie Mellon, Silicon Valley, http://www.cloudcommons.com/about-smi.

[5] S.K. Garg, S. Versteeg, R. Buyya, "SMICloud: Aframework for Computing and Ranking Cloud Services", in Proceedings of the Fourth IEEE International Conference on Utility and Cloud Computing, 2011, pp. 210-218.

[6] He Q., Han J., Yang Y., Grundy J., Jin Hai, "QoS-Driven Service Selection for Multi-Tenant SaaS", in Proceeding of the fifth International Conference on Cloud Computing, pp. 566-573, 2012.

[7] S. Sundareswaran, A. Squicciarini, D. Lin, " A Brokerage-Based Approach for Cloud Service Selection", in Proceedings of the 2012 IEEE Fifth Internatinal Conference on Cloud Computing, 2012, pp. 557-565.

[8] T.L. Saaty, "The Analytic Hierarchy and Analytic Network Processes for the Measurement of Intangible Criteria and for Decision-Making", in Multiple Criteria Decision Analysis: State of the Art Surveys, J. Figueira, S. Greco, and M. Ehrgott, Ed., Springer Verlag, Boston, 2005, pp. 345-405

[9] D. Lin, C. Shi, T. Ishida, " Dynamic Service Selection Based on Context-Aware QoS", in Proceedings of 2012 IEEE Ninth International Conference on Service Computing, 2012, pp. 641-648.

[10] R. Karim, C. Ding, "An Enhanced PROMETHEE Model for QoS-Based Web Service Selection", in Proceedings of the Eighth International Conference on Services Computing, 2011, pp. 537-543.

[11] K. Kritikos, and D. Plexousakis, "Mixed-Integer Programming for QoS-based Web Service Matchmaking", IEEE Transactions on Services Computing, vol.2, no.2, 2009, pp.122-139.

[12] C. Hang, M. Singh, "From Quality to Utility: Adaptive Service Selection Framework", in Proceedings of the Eighth International Conference on Service-Oriented Computing, CA, USA, 2010, pp. 456-470.

[13] L.D. Ngan, R. Kanagasabai, " OWL-S Based Semantic Cloud Service Broker", in Proceedings of the IEEE 19th International Conference on Web Services, 2012, pp. 560-567.

[14] T. Heninger , A. Singh, V. Singh, T. Wies, D. Zufferey , "A Marketplace for Cloud Resources", in Proceedings of the International Conference of Embedded Systems, 2010, pp. 1-7.

[15] O. Krieger, P. McGachey, A. Kanevsky, "Enabling a Marketplace of Clouds: VMware's vCloud Director", OPERATING SYSTEMS REVIEW, Vol. 44, Issue 4, 2010, pp. 103-114.

[16] A. Menychtas *et al*, "A Marketplace Framework for Trading Cloud-Based Services", in Proceedings of the 8th Workshop on Economics of Grids, Clouds, Systems, and Services, 2011, pp. 76-89.

[17] Z. Mammeri, "Approach for End-to-End QoS mapping and Handling", in Proceedings of the 2nd International Conference on Wireless and Optical Communications Networks, 2005.

[18] E. Cerqueira, *et al*, " QoS Mapping and Adaptation in Next Generation Networks", in Proceedings of the International Symposium on Applications and the Internet Workshops, 2007.

[19] S. B. Rakas, M. D. Stojanovic, " An Efficient QoS Mapping Algorithm in Multi-Provider Networks", in Proceedings of the 34th International Conference on Telecommunications and Signal Processing (TSP), 2011, pp. 74-78.

[20] T. Yu, K. Lin, " A Broker-Based Framework for QoS-Aware Web Service Composition", in Proceedings of the IEEE International Conferences on e-Technology, e-Commerce and e-Service, 2005, pp. 22-29

[21] Q. Ma, H. Wang, Y. Li, G. Xie, and F. Liu, "A Semantic QoS-aware Discovery Framework for Web Services", in Proceedings of the IEEE International Conference on Web Services, 2008, pp.129-136.

[22] E. Bauer,R. Adams, R."Service Reliability and Service Availability", in Reliability and Availability of Cloud Computing, John Wiley & Sons, Inc., Hoboken, NJ, USA, 2009. doi: 10.1002/9781118393994.ch3.

[23] S.W. Choi, J.S. Her, S.D. Kim, "QoS Metrics for Evaluating Services from the Perspective of Service Providers", IEEE International Conference on e-Business Engineering, China,2007, pp.622-625.

[24] M. Armbrust, *et al*, "Above the Clouds: A Berkeley View of Cloud Computing", Technical Report No. UCB/EECS-2009-28.