An Adaptive One-to-many Negotiation to Improve The Service Acceptability of an Open SaaS Provider

Amro Najjar
Hubert Curien Laboratory
Saint-Etienne, France
amro.najjar@emse.fr

Olivier Boissier
Hubert Curien Laboratory
Saint-Etienne, France
olivier.boissier@emse.fr

Gauthier Picard
Hubert Curien Laboratory
Saint-Etienne, France
gauthier.picard@emse.fr

ABSTRACT

Service acceptability rate and user satisfaction are becoming key factors to avoid client churn and secure the success of any Software as a Service (SaaS) provider. Nevertheless, the provider must also minimize the cost it pays to rent resources from the cloud. To address these contradicting concerns, most of existing works carry out resource management unilaterally by the provider. Consequently, end-user preferences and her subjective acceptability of the service are mostly ignored. In order to assess user satisfaction and service acceptability, Mean Opinion Score (MOS) used to be the most popular metric. However, recent studies in the domain of Quality of Experience (QoE) recommend providers to use better metrics such as quantiles to gauge user service acceptability more precisely. In this article we propose an adaptive one-to-many negotiation mechanism to improve the service acceptability of an open SaaS provider. Based on quantile estimation of service acceptability rate and a learned model of the user negotiation strategy, this mechanism adjusts the provider negotiation process in order to guarantee the desired service acceptability rate while meeting the budget limits of the provider. The proposed mechanism is implemented and its results are examined and analyzed in light of comparable results.

CCS CONCEPTS

•Computing methodologies → Multi-agent systems;

KEYWORDS

Adaptive one-to-many negotiations, acceptability rate, SaaS, Cloud Computing

ACAN Workshop @ AAMAS:
Amro Najjar, Olivier Boissier, and Gauthier Picard. 2017. An Adaptive One-to-many Negotiation to Improve The Service Acceptability of an Open SaaS Provider. In *Proceedings of , Sao Paulo, Brazil, May 2017 (ACAN 2017 Workshop)*, 9 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Client churn is one of the most negative indicators effecting any online SaaS provider. A recent study of customer dynamics by Accenture showed that the majority of consumers in USA switched their service provider because it fails to meet their expectation and that about 81% of these customers said that the company could have done something differently to prevent them from switching [1]. Yet, user satisfaction and client churn are not the only challenges providers need to overcome in order to remain in the business, the popularity of multimedia-rich applications are expected to rise signficiantly [14]. Therefore, tomorrow's market is shaped by intensive but fluctuating demand and high user expectations. In order to cope with this rapid evolution, Application Service Providers (ASP) are increasingly migrating to the cloud to manage their resources in elastic manner and thereby minimizing their operational costs.

Thus, the ASP or the SaaS provider has to balance two concerns: minimizing client churn while meeting its budget constraints. In the context of cloud computing this issue is known as elasticity management or auto-scaling [7, 26] and it has received considerable attention in the recent years. However, most of the works tackling this issue adopt a centralized approach where the ASP takes the resource decision unilaterally [7, 26]. Consequently, the end-user preferences are mostly overlooked and it is often presumed that their acceptability threshold tolerates the best-effort service proposed by the provider.

Multi-agent systems have been outlined as a platform to account for the end-user satisfaction [25, 27] and achieve essential tasks such as resource allocation and coordination in this multi-partite ecosystem [42]. One-to-many multi-agent negotiation provides a potential platform to involve the end-user into the elasticity management process. Nevertheless, in contrast to the case of a SaaS provider, where the goal is to maximize the acceptability rate by reaching as much agreements as possible, the majority of existing works in the literature addresses a scenario where the seller seeks to find one *atomic* agreement reached with one of the concurrent buyers. Furthermore, most of these works assume a *closed* set of participants in which the users are assumed to be known in advance before the outset of the negotiation process. This assumption does not hold in today's online and cloud ecosystem where hundreds of users may enter the negotiation process every minute.

In this article we develop novel adaptive and open multi-issue negotiation and coordination mechanisms allowing the provider to achieve a targeted service acceptability rate while satisfying its budget constraints. Users can decide whether to accept or reject the proposed service depending on their expectations and subjective estimation of the service quality [25]. Based on its measurements of the portion of users finding the service unacceptable, the provider adjusts its negotiation strategy in order to restore the acceptability rate to its predefined goals. Thus, using the proposed mechanism, the SaaS provider can *(i)* accommodate the dynamic and open nature of the cloud ecosystem where users may enter the system, negotiate, and leave at will *(ii)* integrate the user preferences into the decision process to reach mutually-accepted agreements hereby ensuring a precise predefined acceptability rate and meeting its business objectives.

ACAN 2017 Workshop, Sao Paulo, Brazil
2017. 978-x-xxxx-xxxx-x/YY/MM…$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

The proposed negotiation and coordination mechanisms are developped in the EMan architecture[24].

The rest of this article is organized as follows: Section 2 offers a background on service acceptability and Quality of Experience (QoE) then shows how one-to-many negotiation can provide a potential solution allowing the provider to satisfy its business goals and integrate the user into the decision process. Section 3 reviews the EMan architecture, its agents, its negotiation and coordination protocols. Section 4 details the adaptive negotiation strategy and the proposed opponent learning and modeling approach. Section 5 details the evaluation process and discusses the results. Section 6, discusses the related works. Finally, Section 7 concludes this paper and points out future research perspectives.

2 MOTIVATION

Service acceptability and client churn are tightly related to the user's subjective perception of the service quality [11, 18] or the *Quality of Experience* [23]. In Section 2.1 we discuss the shortcomings of existing user satisfaction estimation metrics and review the latest research in the domain of QoE where recent works propose *better* metrics to gauge service acceptability rate [11, 12]. Then, Section 2.2 discusses one-to-many negotiations and their potential in cloud elasticity management process.

2.1 QoE and Service Acceptability

Quality of Experience (QoE) or the service quality as perceived by the user is known to be a key determinant of the user's decision to accept or reject a service [11, 23].

Although most of studies dealing with QoE are carried out on the conceptual front (because the field is in its early stages of development), QoE is targeted to provide a practical measure allowing to quantify user satisfaction and acceptance of the service [45]. In particular, QoE-management emerged as a process aiming at maximizing QoE while optimizing the used resources [37]. However, most of existing works suffer from the following two important limitations.

First, QoE-management literature largely relies on the Mean Opinion Score (MOS) to assess satisfaction and service acceptability [40] and it has been used both for Next Generation Networks (NGN) applications and cloud computing. However, since it is an average of users' opinions, MOS hides important information about user diversity and their personal preferences [10]. Yet, this difference in opinions can have significant impact for the provider. For this reason, other measures have been proposed to allow the provider to understand the end-user satisfaction and estimate client churn [11]. For instance, quantiles and percentiles have been proposed as a measurement tool allowing the provider to ascertain that, say, 95% of its users find the service to be acceptable or better [11, 12].

Second, in the majority of these works (both in NGN and Cloud Computing), QoE-Management remains a unilateral process achieved by the provider [37] without integrating the end-user preferences into the loop. Even when monitoring agent are distributed across the network, the decision making process is undertaken by the provider (e.g. [17]). Therefore, despite the declared goal of establishing a subjective and user-centric estimation of service quality, existing works are predominantly provider-centric.

2.2 Multi-Agent One-to-many Negotiation for Elasticity Management

Elasticity and cloud resource management have received considerable attention since the emergence of cloud computing. Several works in the literature have formulated elasticity management as a constraint satisfaction problem whose goal is to maximize a global utility function defined by the provider while meeting the budget constraints [5]. However, these problems are inherently NP-hard. Moreover, they presuppose that the user requests and the workload applied to the system are known in advance. Consequently, several heuristic policies have been proposed to overcome these limitations (c.f. [5] and the references therein). Nevertheless, in most of these works the end-user preferences and their subjective service acceptability are either overlooked or assumed to be known in advance by the provider. Hence, the elasticity management process is done unilaterally by the provider.

By definition, an agent is usually self-interested and is bound to an individual perspective [47]. This makes agents potential candidates to represent the subjectivity of users' opinions and acceptance of a given service [27]. Furthermore, the principles of negotiation behavior discussed in [30] provide a useful tool to represent end-user expectations. The latter are a key determinants of user satisfaction and service acceptability [35, 48] (for more about this discussion please refer to [25]). In addition, multi-agent systems provide a distributed platform capable to undertake tasks that require cooperation such as resources management [42].

One-to-many negotiation is a sub-type of multi-agent negotiation [19] where one agent (e.g. a seller) negotiates simultaneously with multiple agents (e.g. potential buyers). In multi-agent negotiation literature, several approaches have been proposed to tackle this type of negotiation. Even though they address different applications, the majority of these solutions (e.g. [2, 20, 31]) share a common architectural blueprint. Yet, the negotiation and coordination strategies used in each solution depends on its purpose [24].

One-to-many multi-agent negotiation is a potential solution to integrate the end-user and her subjective service acceptability into the elasticity management. Nevertheless, in contrast to the case of a SaaS or online provider where the goal is to maximize the acceptability rate by reaching as much agreements as possible, the majority of existing works in the literature of one-to-many negotiations addresses a scenario where the seller seeks to find one *atomic* agreement reached with one of the concurrent buyers. Furthermore, most of these works assume a *closed* set of participants in which the opponents of the single agent are known in advance before the outset of the negotiation process. Moreover, in most of the existing works, offers are sent and received in a synchronous manner. These assumptions do not hold in today's open cloud ecosystem where hundreds of users may enter/leave the system every minute and where the negotiation sessions are not synchronized. Section 6 provides further discussions about the related works in one-to-many negotiations and their limitations.

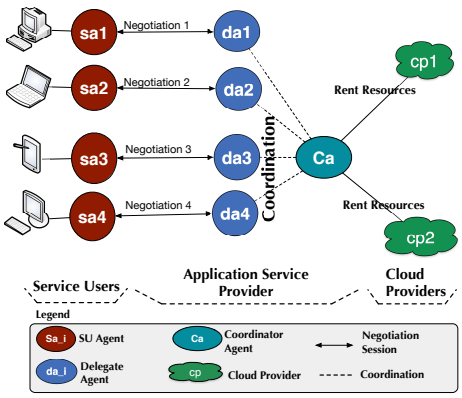

Figure 1: The EMan architecture deployed in the cloud ecosystem.

3 THE ELASTICITY MANAGEMENT ARCHITECTURE (EMAN)

In this article we present a novel one-to-many negotiation mechanism that meets the following objectives:

(1) *Adaptive*: whenever one of the business objectives of the provider is violated (e.g. the targeted acceptability rate), the mechanism enables the provider to adjust its negotiation behavior in order to restore the desired level while still maintaining other objectives (e.g. a budget limit).
(2) *Open* set of participants & *Non Synchronous* management: to accommodate the dynamic & open nature of the cloud ecosystem, the proposed mechanism allows user agents to enter and leave the system whenever they desire. Furthermore, the negotiation process is non-synchronous. Some negotiation sessions are already over, others are still on going, and others have not started yet.

The proposed mechanism is implemented in the EMan architecture [24, 25]. The latter is multi-agent arhitecture for SaaS elasticity management. EMan (Figure 1) follows the same architectural blueprint discussed in Section 2.2 and shared by most of existing one-to-many solutions in the literature. In the earlier version of EMan [24, 25], users are represented by autonomous agents whose goal is to maximize the QoE of their respective users. However, these earlier versions do not include the adaptive mechanism proposed in this article.

This section introduces the types of agents involved in the EMan architecture (Section 3.1), presents the bilateral negotiation process (Section 3.2) and discusses the role assumed by the coordinator (Section 3.3).

3.1 Agents

The EMan architecture depicted in Figure 1 models the negotiation taking place between a SaaS provider and its end-users or Service Users (SU). The negotiation between the SaaS providers and the cloud providers is considered beyond the scope of this article. The EMan architecture contains three types of agents: service user agents (denoted as sa_i), delegate agents (denoted as da_i) and a single coordinator (denoted *ca*). The latter two types represent the provider. These agents are introduced in the following subsections respectively.

3.1.1 Service User Agents. A service user agent participates in the negotiation process on behalf of a service user. a sa_i has a utility function M_{sa_i} that encodes its preferences. M_{sa_i} is used at each cycle t to assess the utility of offers $o^t_{da_i}$ received from the corresponding delegate (see Section 3.2.1). If the service involves J attributes, then M_{sa_i} is defined as follows:

$$M_{sa_i}(o^t_{da_i}) = \sum_{j=1}^{j=J} w_{sa_i,j} \cdot \mu_{sa_i,at_j}(o^t_{da_i}[at_j]) \quad (1)$$

Where w_{sa_i,at_j} is the weight associated with attribute at_j to specify how much importance this user gives to this attribute (these weights should satisfy $\sum_{j=1}^{J} w_{sa_i,at_j} = 1$), $o^t_{da_i}[at_j]$ is the value offered in $o^t_{da_i}$ for the j^{th} attribute (*i.e.* at_j), and μ_{sa_i,at_j} is the utility function of this attribute. This function is defined as follows:

$$\mu_{sa_i,at_j}(o^t_{da_i}[at_j]) = \frac{rv_{sa_i,at_j} - o^t_{da_i}[at_j]}{rv_{sa_i,at_j} - pv_{sa_i,at_j}} \quad (2)$$

Where $\mu_{sa_i,at_j}(o^t_{da_i}[at_j]) \in [0,1]$, and pv_{sa_i,at_j} and rv_{sa_i,at_j} are respectively the preferred (*i.e.* best) and reservation (*i.e.* worst) values of sa_i for this attribute [1]. Note that, as has been confirmed by empirical studies, the user is capable of choosing the values of pv_{sa_i,at_j}, rv_{sa_i,at_j} and $w_{sa_i,j}$ [25, 35, 48]. Therefore, sa_i can obtain them from the user. However, these values are not divulged to the provider.

In order to make an accept/reject decision, a sa_i relies on its utility function and on its current Aspiration Rate (AR). $AR^t_{sa_i}$ expresses how much utility sa_i expects to obtain in this negotiation cycle t. $AR^t_i \in [0,1]$. When $AR^t_i = 1.0$, this means that a_i expects obtaining its preferred values for each attribute. On the other hand, $AR^t_i = 0.0$ indicates that the agent is ready to accept the worst possible offer. In order to reach agreements, sa_i makes concessions by reducing its AR. All the sa_i follow a Time-Based Concession strategy (TBC) [9]. This assumption is quite common in the literature notably in works whose goals is to constuct a model of the opponent behavior (c.f. [4]). Therefore, $\Delta AR^t_{sa_i}$, the concession made by sa_i for the cycle t, depends on the time left before reaching the deadline. It is computed as follows [39]:

$$\Delta AR^t_{sa_i} = AR^{t-1}_{sa_i} \cdot \left(\frac{t}{T_{sa_i}}\right)^{\lambda_{sa_i}} \quad (3)$$

Where T_{sa_i} is the negotiation time deadline. λ_{sa_i} is a parameter that controls the convexity degree of sa_i's concession curve . λ_{sa_i} determines the behavior of sa_i (conciliatory, linear or conservative) [39].

[1] As discussed in [24, 25], μ_{sa_i,at_j} may also be a logarithmic function derived from the Weber-Fechner Law [44] and the logarithmic hypothesis [32]. However, the results of the adpative mechanism discussed in this article hold for both linear μ_{sa_i,at_j} (as in Equation 2) and logarithmic functions.

3.1.2 Delegate Agents. Once a new user agent sa_i enters the system, a new delegate da_i is created and it enters a bilateral negotiation session with sa_i.

Like sa_i, a da_i has a utility function M_{da_i}. However, the utility of an offer $o^t_{sa_i}$ from a delegate da_i standpoint is determined by the cost required to serve the offer. M_{da_i} is defined as follows:

$$M_{da_i}(o^t_{sa_i}) = \frac{RC - Cost(\Pi(o^t_{sa_i}))}{RC - PC} \tag{4}$$

Where Π is used by the delegate to estimate the amount of resources needed to serve a given offer $o^t_{sa_i}$, whereas *Cost* computes the cost of the required resources. Thus, $\Pi(o^t_{sa_i})$ estimates the resources needed to realize $o^t_{sa_i}$, *Cost* is the cost function, and $M_{da_i} \in [0, 1]$. PC, RC are respectively the preferred (minimal) and the reservation (maximal) cost of the delegate da_i. These values are initialized by the coordinator when da_i is spawned. Note that the value of RC is very important since it allows the provider to impose its budget constraint *i.e.* the average cost spent on a user should not exceed RC.

If the offer is not accepted by da_i, then it uses its negotiation strategy to generate a counter-offer. In order to reach agreements with the sa_i, as for sa_i, da_i use time-based concession strategy that reduces its aspiration rate (AR). The concession is computed as follows: $\Delta AR^t_{da_i} = \frac{1}{T_{da_i}}$. Where T_{da_i} is the time deadline of the delegate. It is initialized by the coordinator based on its business knowledge. When da_i reaches T_{da_i}, it does not quit the negotiation process, it will just stop making more concessions.

The EMan architecture contains a single coordinator agent, its role and scope of intervention are discussed in Section 3.3 where we explain the used coordination strategies.

3.2 SU-Delegate Bilateral Negotiation

In the EMan architecture, da_i and sa_i do not have access to the preferences and negotiation strategies of other agents. Therefore, their negotiation behavior follows the *negotiation decision function* [9] and it is determined by the utility function and negotiation strategy. As for the coordinator, it is capable of manipulating the negotiation strategies of the delegates if the adaptation mode is active.

3.2.1 Negotiation Object. The negotiated service is characterized by a set of J attributes $s = \langle at_1, at_2, \cdots, at_J \rangle$.

The object o^t_i is the offer/counter-offer exchanged during the negotiation process at step t taking place between sa_i and da_i. It assigns a value v^t_k to each one of the attributes describing the negotiated service: $o^t_i = \langle v^t_1, v^t_2, \cdots v^t_J \rangle$.

The negotiation between a da_i and a sa_i is based on the alternate offer protocol [34]. When a sa_i receives an offer it can either (*i*) accept the offer, (*ii*) propose a counter-offer or (*iii*) quit the negotiation process. Once an offer is accepted by an agent, it cannot renege it.

In order to accommodate the open and dynamic nature of the cloud ecosystem, negotiation sessions in the EMan architecture are non-synchronous *i.e.* some sessions will be already terminated while other sessions will be still active or have not started yet. For further information about the negotiation protocol and the acceptance strategy please refer to [24, 25].

3.3 Coordination Strategies

The coordinator, denoted as *ca*, is defined by the following tuple:

$$ca = \langle \Pi, Cost, \Omega_{initial}, AcceptRate, Surplus \rangle \tag{5}$$

Where Π and *Cost*, are the performance model and the cost function introduced in Section 3.1. The provider builds them based on its business knowledge. $\Omega_{initial}$ is the delegate initial negotiation strategy and it is used to initialize delegates when they are spawned by the *ca*. *AcceptRate* is the acceptance rate *i.e.* the percentage of users who accepted the service to all users who entered the system up till a given moment. *Surplus* is a variable where the ASP accumulates all the surplus obtained from successful sessions.

In the EMan architecture, as long as the ASP business goals are not violated (a coordinator intervention case detailed in Section 4), delegates assume the bilateral negotiation with the sa_i in an autonomous manner. Yet, delegates solicit the coordinator when a negotiation session is terminated either successfully or unsuccessfully. The following sections list these cases.

3.3.1 Successful Negotiation Session. When a delegate da_i reaches agreement with the corresponding sa_i, it notifies the coordinator. In order to estimate the acceptance rate, the coordinator keeps track of successful negotiation sessions. Furthermore, when a session is declared successful, the coordinator updates its *Surplus* variable as follows: $Surplus = Surplus + surplus_i$, where $surplus_i$ is the surplus obtained from the successful negotiation session i. $surplus_i$ is computed as follows:

$$surplus_i = RC - Cost(\Pi(\hat{o_i})) \tag{6}$$

Where RC is the reservation cost of the delegates. $\hat{o_i}$ is the accepted offer, and $Cost(\Pi(\hat{o_i}))$ is the cost to be paid by the ASP to satisfy this offer. Thus, the difference between the actual cost and the reservation (maximum) cost is considered as surplus.

3.3.2 Failed Negotiation Session. Whenever a delegate da_i observes that the corresponding sa_i has left the negotiation session, da_i notifies the coordinator. The latter updates the *AcceptRate* variable in order to ascertain that the targeted acceptance rate (e.g. 95% of users) is met. Otherwise, the coordinator triggers the adaptation process. This process is detailed in the next section.

This section offered an overview the EMan architecture proposed in [24, 25]. We are aware that the negotiation protocols and strategies presented above can be exploitable. However this issue is beyond the scope of this article. We consider this issue to be of lesser importance for the following reason: the negotiation in cloud ecosystem have a strong *integrative* component since the resources offered by the cloud appear to be unlimited (*i.e. expendable pie* negotiation c.f. [43]). Therefore, instead of exploiting the users' negotiation strategies and force them into making hefty concessions, the provider is more likely to seek win-win settlements allowing to maximize the acceptability rate while meeting its budget constraints.

The following section details the adaptive algorithm proposed in this current work.

4 ADAPTIVE NEGOTIATION STRATEGY

This section details the adaptive algorithm. In order to estimate the current acceptance rate at any given time t, the provider relies

on a quantile estimation algorithm presented in Section 4.1. If the targeted acceptance rate is violated, the coordinator activates the adaptive mode. With this mode active, a delegate da_i has to analyze the behavior of its opponent sa_i, learn a model of its negotiation strategy in order to estimate its negotiation time deadline $\bar{T_{sa_i}}$, and send this estimation to the coordinator (explained in Section 4.2). The latter chooses the high-priority sessions (based on their time deadlines) and adjust their negotiation strategies while respecting its budget constraints (Section 4.3).

4.1 Triggering The Adaptation Mechanism

Quantiles and percentiles are values that partition a finite set of values into q subsets of (nearly) equal sizes. As discussed in Section 2.1, the literature on QoE and user satisfaction recommend providers to rely on quantiles and percentiles as more accurate measures (compared with MOS) to gauge user acceptability of the service [12].

Whenever session i is terminated, the coordinator is notified by da_i about the outcome of this session. Using these data, the coordinator runs a quantile estimation algorithm to detect the current service acceptability rate.

Let Q the quantile/percantile estimation function. Let R be the dataset containing the outcomes of the terminated sessions. R can contain either 0's, for failed sessions, or 1's for successful sessions. If the coordinator seeks to ensure that β percent of users who requested the service so far have had a successful negotiation session (hereby accepting the proposed service quality), the coordinator needs to verify that the $(100 - \beta + 1)$th percentile equals 1:

$$Q(R, 100 - \beta + 1) = 1 \tag{7}$$

As long as this condition holds, the coordinator has no need to intervene into the negotiation process. To calculate Q we rely on [6] where the authors develop an algorithm for effective computation of quantiles over data streams.

Once the condition in Equation 7 is violated, the coordinator triggers the adaptation mechanism by commanding all working delegates to activate their adaptive mode. Furthermore, when the coordinator spawns new delegates they will have this mode active as well.

Note that the coordinator continues to evaluate Equation 7 even after the activation of the adaptive mode. Next section details the delegate's role after the activation of the adaptive mode.

4.2 Opponent Learning and Modeling Algorithm

4.2.1 Opponent Concession Estimation. Even when the adaptive mode is not active, when a delegate da_i receives an offer $o^t_{sa_i}$ at cycle t from the corresponding sa_i it estimates the concession made by sa_i by comparing $o^t_{sa_i}$ with $o^{t-1}_{sa_i}$ the previous offer made by sa_i. Since da_i does not have access to sa_i preferences or utility function M_{sa_i}, it cannot calculate the real concession made by sa_i. Instead, it relies on its own utility function to estimate the concession made by sa_i by assuming that a concession made by sa_i is synonymous with a utility gain for da_i. Thus, $c^t_{sa_i}$, the estimated concession made by sa_i at the negotiation cycle t is defined as:

$$c^t_{sa_i} = M_{da_i}(o^t_{sa_i}) - M_{da_i}(o^{t-1}_{sa_i}) \tag{8}$$

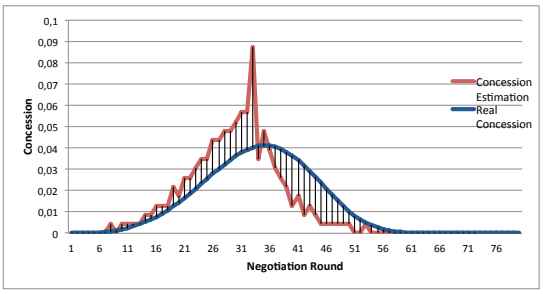

Figure 2: $\Delta AR^t_{sa_i}$, the real concession made by sa_i in blue compared with $c^t_{sa_i}$ da_i's estimation of the same concession (in red).

To learn sa_i concession behavior, da_i keeps track of sa_i concessions during the negotiation session. Figure 2 compares $\Delta AR^t_{sa_i}$ the real concessions made by sa_i (in blue) with $c^t_{sa_i}$, da_i's estimation of the same concession (in red). In this example, $T_{sa_i} = 80$ cycles and $\lambda_{sa_i} = 3.0$. As can be seen from the figure, although the red curve is noisy (because it is estimated using da_i utility function and then normalized), it seems to provide an adequate estimation of sa_i concession since the main goal is to learn T_{sa_i}, the negotiation time deadline of sa_i.

4.2.2 Opponent Model Learning. If the adaptation mode is active, the concession data collected in Section 4.2.1 should be used to establish a model of sa_i concession behavior and infer T_{sa_i}, its negotiation time deadline. This can be achieved, as has been shown in the literature [4], using non-linear regression assuming that all users follow a time-based concession defined by Equation 3. Yet, unlike the predominantly one-to-one existing works in the literature [4] where learning the negotiation model of the opponent is done and corrected each cycle once a new concession is made (c.f. Section 6), in the one-to-many negotiation settings addressed by this article such an approach is not practical for scalability reasons since hundreds or thousands of users may be negotiating with the provider simultaneously. Furthermore, if the provider runs delegates' learning algorithms on resources rented from the cloud, this may increase the costs considerably.

For this reason, in the proposed solution a delegate can run the non-linear regression algorithm only once. The next decision is to determine when a delegate should do so. On the one hand, if a delegate launches this process too early in the negotiation session, the regression process will have only few input points. On the other hand, if the delegate waits too long, the corresponding user risks to reach its time deadline and quit the negotiation process.

All sa_i follow time-based concession strategy in which the rate of concession sa_i slows down when sa_i has already made most of the concessions it was going to make. Therefore, regardless of the type of sa_i (conservative or linear, c.f. Equation 3) and of its actual time deadline T_{sa_i}, when the rate of change (or the derivative) of da_i's estimation of sa_i concession decays significantly into negative values, this means that sa_i has made most of its concessions and that, for the rest of the negotiation session before sa_i reaches its T_{sa_i}, sa_i will stop making considerable concessions. Thus, if da_i

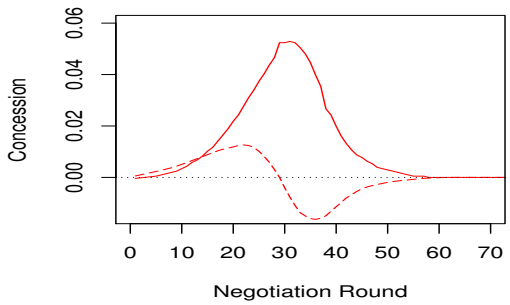

Figure 3: A smoothed version $c^t_{sa_i}$ (solid line) and its derivative (dashed line).

launches its non-linear regression at this stage, it will have the most of useful data points.

Figure 3 superimposes da_i's estimate of the concession made by sa_i (red solid line) with its first derivative (red dashed line). The sa_i in this figure is exactly the same of Figure 2, but we used a Savitzky-Golay filter [36] to smooth the curve and filter out the noise. To calculate the rate of change of the concessions, we calculate the derivative and we smooth the result using the a variant of the same filter.

The non-linear regression algorithm takes the negotiation cycle (t) and the estimated concession ($c^t_{sa_i}$) as input variables and outputs an estimated values of the $\bar{T_{sa_i}}$ and $\bar{\lambda_{sa_i}}$ parameters. Then, based on $\bar{T_{sa_i}}$, da_i computes $\bar{r_i} = \bar{T_{sa_i}} - t$ the estimated number of cycles remaining in session i. As long as the session i is not terminated, da_i updates $\bar{r_i}$ every cycle and sends it repeatedly to the coordinator.

4.3 Negotiation Adaptation

As was explained in the previous section, the coordinator receives $\bar{r_i}$ from all negotiation sessions i whose delegate da_i considers that sa_i is approaching its time deadline and deserves to be prioritized. These estimations $\bar{r_i}$ are stored in the *priority list* which is continuously sorted in ascending manner (a session whose time deadline is estimated to be sooner will be in the top of the list).

Note that since negotiation sessions are non-synchronized, the coordinator (ca) has to repeat the sorting whenever it receives a new estimation from a delegate da_i. Furthermore, whenever a session is terminated either successfully or not, ca removes its record from the priority list.

Thus, in the priority list sessions are sorted from highest to lowest priorities. Now ca must decide how many of these priority sessions should receive a preferential treatment. To do so, the coordinator undertakes this process as follows:

(1) Estimate the current acceptability rate (using the quantile estimation function as discussed in Section 4.1). Then compute the difference between the desired acceptability rate and its actual current value.
(2) Estimate p the number of successful sessions required to restore the desired rate and choose the first p sessions from the priority list and adjust their negotiation strategies to encourage them accept the service.

When the negotiation strategy of a delegate da_i is adjusted, da_i retains the a time-based concession strategy defined in Section 3.1.2. Yet, with the following modifications.

First, the negotiation time deadline of da_i becomes $\bar{r_i}$ instead of T_{da_i}. Second, the reservation cost of da_i is increased by value denoted as *RcPrio*. *RcPrio* is computed by dividing the surplus available in *Surplus* among all the prioritized sessions. Note that delegates cannot get more than a $RcPrio_{max}$ considered as the maximum value for prioritized sessions. Thus, the provider ensures the satisfaction of its budget constraints. Third, the preferred cost of da_i is changed to the cost of the last offer it has made. $AR^t_{da_i}$ is restored to 1.

This section presented the opponent modeling and the negotiation adaptation processes. Next section presents the experimental evaluation.

5 EVALUATION

To evaluate the adaptive mechanism, we implement it in the EMan architecture ([24, 25]). The latter is a multi-agent architecture for SaaS elasticity management implemented using Repast Simphony [29], a multi-agent simulation environment. The evaluation is organized into three experiments. The first experiment (Section 5.2) aims at evaluating the adaptation algorithm. The second experiment (Section 5.3) examines the impact of the workload applied to the provider (i.e. the number of user entering the system per minute) on the acceptability rate. The third experiment (Section 5.4), evaluates the overhead of the negotiation/coordination mechanisms. Before discussing these experiments, next section presents their parameters.

5.1 Experiments Parameters

The total number of users entering the system is denoted as $|SU|$. In the following experiments, $|SU| = 10000$ users. Users enter the simulation following a Poisson random process whose mean value is A per minute. The service in the experimental scenario involves two attributes: one is the service delivery time while the other represents the service quality. The user profiles (reservation, preferred values and weights for each attributes) are generated randomly. The negotiation time deadlines of sa_i are generated randomly $T_{sa_i} \in [40 : 120]$, and $\lambda_{sa_i} \in [1 : 8]$ (*i.e.* users can be linear or conservative). The cost of services acceptable by users ranges from 0.1 $ to 0.9$. RC, delegate reservation cost (the maximum cost allocated to a non-prioritized user) is set to 0.60$. This parameter represents the provider's budget constraints. Therefore, without the adaptation mode, approximately a quarter of users will not accept the service since RC cannot satisfy their least expected service. $RcPrio_{max} = RC/2$. Therefore, when a user sa_i is prioritized, the RC of the respective delegate (da_i) is at most increased by $RC/2$. *Goal* is the percentage of users that the provider seeks to satisfy. Its impact is evaluated in the next subsection. Note that the results discussed below remain valid even when the values of the parameters above (e.g. RC, $RcPrio_{max}$, etc.) are changed. Furthermore, we obtained similar results when user attribute utility function is a logarithmic function derived from the logarithmic hypothesis [25, 32].

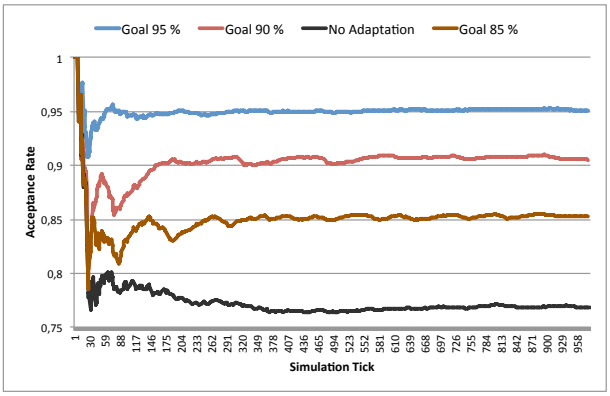

Figure 4: Acceptance Rate with $A = 160$.

Table 1: The impact of the workload parameter A.

$A =$	20	40	80	160	320	640	1280	2500
Adaptation	95%	95%	95%	95%	95%	94.9%	94.7%	93.3%
No-Adaptation	77%	77%	77%	77%	77%	77%	77%	77%

5.2 Evaluating The Adaptation Mechanism

Figure 4 shows the results of this experiment. The blue curve draws the acceptance rate when the *Goal* = 95%. As can be seen from the figure, at the outset of the simulation, the value of the acceptance rate oscillated before getting stabilized on the goal value (*Goal* = 95%). The red and the brown curves, that plot the acceptance rate when *Goal* = 90% and *Goal* = 85%, show similar results. To compare the results of the adaptation mechanism, the black curve plots the acceptance rate when the mechanism is deactivated.

As can be seen from the results, the adaptation mechanism managed to restore the acceptance rate to the predefined *goal* in a precise manner. This is explained by fact that the prediction algorithm discussed in Section 4.2 managed to predict the user negotiation deadline T_{sa_i} with acceptable precision. The average error rate between the prediction $\tilde{T_{sa_i}}$ and the real value T_{sa_i} is about 6.0 cycles which means a delegate da_i overestimates/underestimates T_{sa_i} by three negotiation cycles on average.

Intuitively, the service acceptability rate achieved by the prediction algorithm comes with more cost invested per user. However, this increase does not violate the provider budget constraint: the average costs per user were 0.55 \$, 0.52\$, 0.5\$ and 0.44\$ for *Goal* = 95%, *Goal* = 90%, *Goal* = 85% and non-adaptive respectively. Thus, the budget constraint of the provider (*i.e.* average cost per user does not exceed RC=0.6 \$) was satisfied because the provider relies on *Surplus* to serve prioritized users.

5.3 The Impact of The Workload A

This experiment studies the impact of A, the workload applied to the system, on the acceptance rate in order to evaluate the elasticity of the adaptation mechanism.

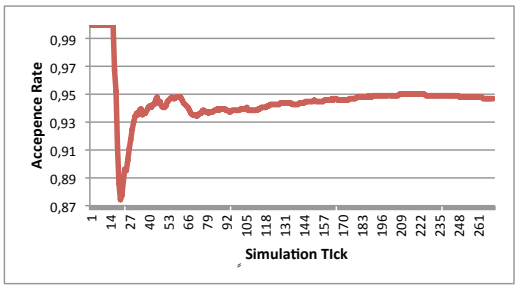

Figure 5: Acceptance rate (*Goal*=95% and $A = 1280$).

Table 1 shows the acceptance rate, at the end of the simulation, when the *Goal* = 95% [2] and with A increasing from 20 to 2500 users per minute. As can be seen from the table, the mechanism proved to be highly elastic. When $A \in [20 : 320]$ the acceptance rate does not change as A increases. However, When A increases to higher values (640 to 2500) the acceptance rate witnesses a slight decrease. However, even with very high A, the results achieved by the adaptation mechanism remain very close to the goal. This decrease is explained as follows: the adaptation algorithm is reactive and the coordinator can only assess the acceptance rate each time a session terminates successfully or unsuccessfully. Thus, when A is too high, a significant number of sessions may fail in the same simulation tick and the adaptation algorithm executed by the coordinator suffers from a slight delay and becomes a bit short from meeting the predefined goal (*Goal* = 95%) as can be seen from the table. Yet, this result should be taken relatively for the following reason:

$|SU|$ the total number of users in the simulation influence the elasticity threshold of the adaptation mechanism. For instance, when $|SU| = 10000$ and $A = 2500$, users enter the architecture in 4 massive waves each carrying roughly a quarter of the total number of users. Thus, the coordinator does not have enough time restore the acceptance rate to (*Goal* = 95%). Figure 5 shows how the adaptation mechanism reacts to a drop in acceptance rate when $A = 1280$. The coordinator manages to restore the value of the predefined goal (*Goal* = 95%). Yet, a new sudden drop occurs with the new wave of users and the simulation is terminated. Note that in real life scenario, when $|SU| = \infty$, the coordinator will be always able to restore the acceptance rate into its goal value even with a significant A.

5.4 Coordination & Negotiation Overhead

To assess the overhead introduced by the coordination and the negotiation mechanisms we count the number of the messages exchanged to undertake interventions of the coordinator and the number of messages exchanged in bilateral negotiation sessions respectively.

Without the adaptive mechanism, the coordinator is solicited only once per session to get the result of the session (success/failure). With the adaptive mechanism, the number of messages exchanged between delegates and the coordinator increases about 50%. We

[2] We conducted the same experiment with different values of $Goal \in \{92\%, 90\%, \text{etc.}\}$ and obtained similar results.

consider this increase to be not significant for the following couple of reasons. First, even with the adaptive mechanism, the tasks assumed by the coordinator are lightweight as has been discussed in Sections 3.3, 4.1 and 4.3. Second, since it is likely that the SaaS runs both the delegates and the coordinator on the same cloud data-center, the cost of communication between them is negligible.

As for the number of negotiation messages, the result show that they witness a slight decrease ($\approx$ 7%) when the adaptive mechanism is active since the latter helps reaching agreements faster.

6 RELATED WORKS

This section discusses related works addressing adaption and learning in multi-agent negotiation (Section 6.1) as well as related works in the domain of one-to-many negotiation Section 6.2.

6.1 Learning Opponent Negotiation Model

Learning and modeling negotiation behavior of the opponent is a mature body of research [4]. The model of the opponent typically used to help an agent adapt to its opponent behavior, reach win-win settlements, or minimize the negotiation cost. Various learning techniques are used in the literature. Some works learn the acceptance strategy of the opponent or its preference profile. More importantly, learning the opponent time deadline has received considerable attention. In particular, some of these works use Bayesian learning techniques [16], while others use non-linear regression ([13]) or use both. Nevertheless, all these works address bilateral negotiation, a choice not applicable for elasticity management where the SaaS provider seeks to maximize the service acceptability rate.

6.2 One-to-many Negotiation

Despite the relatively rich literature of one-to-many negotiations, the goal of most of these works, typically modeling negotiations place between a buyer and numerous competing sellers, is to find a single (*i.e.* atomic) agreement that maximizes the buyer's utility whereas other, less beneficial, sessions are aborted [3, 20, 28, 31, 41, 46]. Furthermore, existing negotiation and coordination mechanisms (e.g. [20, 28, 31]) are predominantly *(i)* closed: participants are identified before the outset of the negotiation process, and *(ii)* synchronous: the coordinator needs to receive all the offers from buyers before making its analysis and commanding delegates to send new offers. Therefore, these solutions cannot accommodate the agile nature of the cloud ecosystem where thousands of SUs may surge the service portal.

Recent works in the domain of service composition use one-to-many negotiation to reach *composite* agreements with more than one (maybe all) sellers of atomic services (e.g [21, 22, 33]) for the sake of bundling a composite service. In [33], the authors propose a one-to-many negotiation mechanism that redistributes the surplus obtained from successful negotiation sessions to ongoing negotiation sessions to increase their chances of success. In [21, 22] Mansour *et al.* develop a more sophisticated approach that adapts delegates reservation values for attributes shared among multiple sessions. Nevertheless, these works assume *closed* set of atomic sellers all known before the outset of the negotiation.

One-to-many negotiation has been used in the domain of cloud computing. The authors in [2] develop a negotiation-based approach to handle resource allocation in cloud computing. However, a provider accepts an offer if and only if it can gain some immediate payoff by accepting the offer. For this reason, user acceptability rate is not taken into account and no adaptive mechanism is proposed. Siebenhaar *et al.* [38] develop a mechanism for concurrent SLA negotiation in cloud-based systems. This work supports composite negotiation. Yet, the work's main contribution seems to be focused on the protocol. Therefore, it does not provide a mechanism to represent user acceptability nor it offers a solution to adjust the delegate negotiation behaviors to account for the objectives of the provider.

In our earlier work [24, 25], we proposed EMan an multi-agent architecture for SaaS elasticity management. EMan uses one-to-many negotiation allowing the SaaS to negotiate simultaneously with multiple users. Each user is represented by a user agent sa_i that seeks to satisfy its preferences and expectations and maximize its QoE. However, those earlier versions of EMan did not allow for adaptive negotiation to improve the acceptability rate of the service.

7 CONCLUSIONS & FUTURE WORKS

This paper presented an adaptive one-to-many negotiation mechanism designed to improve the acceptability rate of a SaaS provider while meeting its budget constraints. The proposed approach endows the provider with a fine-grained control of the desired acceptability rate. Furthermore, as has been shown the results, with the proposed solution the SaaS provider is capable to cope with the dynamic & open nature of the cloud ecosystem and to respond to load spikes in an adequate manner.

Our future research work will be directed towards giving the provider a finer-grained control over the level of user satisfaction it seeks to attain. In particular, the provider should be able to ensure that a predefined percentage of users consider the service to be *Good* or *Better* [12]. Several reports of the ITU [15] and the ETSI [8] recommend moving in this direction (c.f. [12]). However, most of the existing works adopt a provider-centric approach based on the MOS [12]. Our future research works will address this issue. This requires upgrades in the user modeling algorithm and the resulting adaptation mechanism. Another work in progress is allowing service user agents sa_i to use negotiation strategies other than time-based concessions (TBC).

REFERENCES

[1] Accenture. Accenture 2013 global consumer pulse survey global & u.s. key findings, 2013.
[2] B. An, V. Lesser, D. Irwin, and M. Zink. Automated negotiation with decommitment for dynamic resource allocation in cloud computing. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 981–988. International Foundation for Autonomous Agents and Multiagent Systems, 2010.
[3] B. An, K. M. Sim, L. G. Tang, S. Q. Li, and D. J. Cheng. Continuous-time negotiation mechanism for software agents. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 36(6):1261–1272, 2006.
[4] T. Baarslag, M. J. Hendrikx, K. V. Hindriks, and C. M. Jonker. Learning about the opponent in automated bilateral negotiation: a comprehensive survey of opponent modeling techniques. *Autonomous Agents and Multi-Agent Systems*, pages 1–50, 2015.
[5] E. Casalicchio and L. Silvestri. Mechanisms for sla provisioning in cloud-based service providers. *Computer Networks*, 57(3):795–810, 2013.

[6] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava. Effective computation of biased quantiles over data streams. In *21st International Conference on Data Engineering (ICDE'05)*, pages 20–31. IEEE, 2005.

[7] E. F. Coutinho, F. R. de Carvalho Sousa, P. A. L. Rego, D. G. Gomes, and J. N. de Souza. Elasticity in cloud computing: a survey. *annals of telecommunications-annales des télécommunications*, 70(7-8):289–309, 2015.

[8] ETSI. European telecommunications standards institute. http://www.etsi.org/.

[9] P. Faratin, C. Sierra, and N. R. Jennings. Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24(3):159–182, 1998.

[10] T. Hobfeld, R. Schatz, and S. Egger. Sos: The mos is not enough! In *Quality of Multimedia Experience (QoMEX), 2011 Third International Workshop on*, pages 131–136. IEEE, 2011.

[11] T. Hoßfeld, P. E. Heegaard, and M. Varela. Qoe beyond the mos: Added value using quantiles and distributions. In *Quality of Multimedia Experience (QoMEX), 2015 Seventh International Workshop on*, pages 1–6. IEEE, 2015.

[12] T. Hoßfeld, P. E. Heegaard, M. Varela, and S. Möller. Qoe beyond the mos: an in-depth look at qoe via better metrics and their relation to mos. *Quality and User Experience*, 1(1):2, 2016.

[13] C. Hou. Predicting agents tactics in automated negotiation. In *Intelligent Agent Technology, 2004.(IAT 2004). Proceedings. IEEE/WIC/ACM International Conference on*, pages 127–133. IEEE, 2004.

[14] C. V. N. Index. Cisco vni forecast and methodology, 2015-2020. cisco white paper, june 1, 2016, 2016.

[15] ITU. International telecommunications union. https://www.itu.int/.

[16] S.-j. Ji, C.-j. Zhang, K.-M. Sim, and H.-f. Leung. A one-shot bargaining strategy for dealing with multifarious opponents. *Applied intelligence*, 40(4):557–574, 2014.

[17] E. Kafetzakis, H. Koumaras, M. A. Kourtis, and V. Koumaras. Qoe4cloud: A qoe-driven multidimensional framework for cloud environments. In *Telecommunications and Multimedia (TEMU), 2012 International Conference on*, pages 77–82. IEEE, 2012.

[18] H.-S. Kim and C.-H. Yoon. Determinants of subscriber churn and customer loyalty in the korean mobile telephony market. *Telecommunications policy*, 28(9):751–765, 2004.

[19] A. R. Lomuscio, M. Wooldridge, and N. R. Jennings. A classification scheme for negotiation in electronic commerce. *Group Decision and Negotiation*, 12(1):31–56, 2003.

[20] K. Mansour and R. Kowalczyk. A meta-strategy for coordinating of one-to-many negotiation over multiple issues. In *Foundations of Intelligent Systems*, pages 343–353. Springer, 2012.

[21] K. Mansour and R. Kowalczyk. Coordinating the bidding strategy in multiissue multiobject negotiation with single and multiple providers. *Cybernetics, IEEE Transactions on*, PP(99):1–1, 2014.

[22] K. Mansour and R. Kowalczyk. On dynamic negotiation strategy for concurrent negotiation over distinct objects. In *Novel Insights in Agent-based Complex Automated Negotiation*, pages 109–124. Springer, 2014.

[23] S. Möller and A. Raake. *Quality of Experience*. Springer, 2014.

[24] A. Najjar. *Multi-Agent Negotiation for QoE-Aware Cloud Elasticity Management*. PhD thesis, École nationale supérieure des mines de Saint-Étienne, 2015.

[25] A. Najjar, C. Gravier, X. Serpaggi, and O. Boissier. Modeling user expectations satisfaction for saas applications using multi-agent negotiation. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 399–406, Oct 2016.

[26] A. Najjar, X. Serpaggi, C. Gravier, and O. Boissier. Survey of elasticity management solutions in cloud computing. In *Continued Rise of the Cloud*, pages 235–263. Springer, 2014.

[27] A. Najjar, X. Serpaggi, C. Gravier, and O. Boissier. Multi-agent systems for personalized qoe-management. In *Teletraffic Congress (ITC 28), 2016 28th International*, volume 3, pages 1–6. IEEE, 2016.

[28] T. D. Nguyen and N. R. Jennings. Coordinating multiple concurrent negotiations. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1064–1071. IEEE Computer Society, 2004.

[29] M. J. North, T. R. Howe, N. T. Collier, and J. Vos. The repast simphony runtime system. In *Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms. Argonne, Illinois, USA: Argonne National Laboratory*. Citeseer, 2005.

[30] D. G. Pruitt. *Negotiation behavior*. Academic Press, 2013.

[31] I. Rahwan, R. Kowalczyk, and H. H. Pham. Intelligent agents for automated one-to-many e-commerce negotiation. In *Australian Computer Science Communications*, volume 24, pages 197–204. Australian Computer Society, Inc., 2002.

[32] P. Reichl, S. Egger, R. Schatz, and A. D'Alconzo. The logarithmic nature of qoe and the role of the weber-fechner law in qoe assessment. In *Communications (ICC), 2010 IEEE International Conference on*, pages 1–5. IEEE, 2010.

[33] J. Richter, M. Baruwal Chhetri, R. Kowalczyk, and Q. Bao Vo. Establishing composite slas through concurrent qos negotiation with surplus redistribution. *Concurrency and Computation: Practice and Experience*, 24(9):938–955, 2012.

[34] A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica: Journal of the Econometric Society*, pages 97–109, 1982.

[35] A. Sackl and R. Schatz. Evaluating the impact of expectations on end-user quality perception. In *Proceedings of International Workshop Perceptual Quality of Systems (PQS)*, pages 122–128, 2013.

[36] A. Savitzky and M. J. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

[37] R. Schatz, M. Fiedler, and L. Skorin-Kapov. Qoe-based network and application management. In *Quality of Experience*, pages 411–426. Springer, 2014.

[38] M. Siebenhaar, U. Lampe, D. Schuller, R. Steinmetz, et al. Concurrent negotiations in cloud-based systems. In *Economics of Grids, Clouds, Systems, and Services*, pages 17–31. Springer, 2012.

[39] S. Son and K. M. Sim. A price-and-time-slot-negotiation mechanism for cloud service reservations. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(3):713–728, 2012.

[40] R. C. Streijl, S. Winkler, and D. S. Hands. Mean opinion score (mos) revisited: methods and applications, limitations and alternatives. *Multimedia Systems*, 22(2):213–227, 2016.

[41] T. Sun, Q. Zhu, S. Li, and M. Zhou. Open, dynamic and continuous one-to-many negotiation system. In *Bio-Inspired Computing: Theories and Applications, 2007. BIC-TA 2007. Second International Conference on*, pages 87–93. IEEE, 2007.

[42] D. Talia. Clouds meet agents: Toward intelligent cloud services. *IEEE Internet Computing*, (2):78–81, 2012.

[43] L. L. Thompson, J. Wang, and B. C. Gunia. Negotiation. *Annual review of psychology*, 61:491–515, 2010.

[44] L. L. Thurstone. Psychophysical analysis. *The American journal of psychology*, 38(3):368–389, 1927.

[45] I. Wechsung and K. De Moor. Quality of experience versus user experience. In *Quality of Experience*, pages 35–54. Springer, 2014.

[46] C. R. Williams, V. Robu, E. H. Gerding, and N. R. Jennings. Negotiating concurrently with unknown opponents in complex, real-time domains. In *in: Proc. of the Twentieth European Conference on Artificial Intelligence*, pages 834–839, 2012.

[47] M. Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[48] V. A. Zeithaml, L. L. Berry, and A. Parasuraman. The nature and determinants of customer expectations of service. *Journal of the academy of Marketing Science*, 21(1):1–12, 1993.