Received February 14, 2019, accepted March 7, 2019, date of publication March 18, 2019, date of current version April 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2905870

CART, a Decision SLA Model for SaaS Providers to Keep QoS Regarding Availability and Performance

JORDI MATEO-FORNÉS, FRANCESC SOLSONA-TEHÀS, JORDI VILAPLANA-MAYORAL, IVAN TEIXIDÓ-TORRELLES, AND JOSEP RIUS-TORRENTÓ
Department of Computer Science, University of Lleida, 25001 Lleida, Spain
INSPIRES, University of Lleida, 25001 Lleida, Spain

Corresponding author: Francesc Solsona-Tehàs (francesc@diei.udl.cat)

This work was supported in part by the Intelligent Energy Europe (IEE) Programme, in part by the Ministerio de Economía y Competitividad under Contract TIN2017-84553-C2-2-R, and in part by the European Union FEDER through CAPAP-H6 network under Grant TIN2016-81840-REDT. The work done by F. Solsona-Tehàs was supported by the Generalitat de Catalunya.

ABSTRACT Cloud systems are becoming a powerful tool for business. The evidence of the advantages of offering infrastructure as a service (IaaS), platform as a service (PaaS), or software as a service (SaaS) is overwhelming. Therefore, for SaaS providers, it is essential to know the virtual resources required to optimize the service offered and to keep the quality of service (QoS) at the desired levels. This paper presents an analytic model cloud availability and response time (CART) for obtaining the best tradeoff between performance, cost, and availability in a cloud system aimed at providing software as a service. The model aims to guarantee the predetermined availability and response time (RT) agreed with customers in a service-level agreement (SLA) contract while minimizing the cost of the system. A client-transparent error recovery strategy has been considered along with a Poisson traffic model. A sensitivity analysis of the simulated and real workloads is presented to study how a specific SLA influences the cloud service performance regarding the guaranteed RT and availability. The results corroborate the goodness of the model proposed, adjusting the real system accurately. Furthermore, the obtained results provide useful guidelines for cloud or Web-server designers.

INDEX TERMS Cloud computing, availability, response time, service level agreement, quality of service.

I. INTRODUCTION

Over the last decade, cloud computing has revolutionized society and the IT industry [1], [2]. According to the National Institute of Standards and Technology (NIST) [3], Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

In this context, computational resources can be offered to the users as a service (XaaS). The most common are infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS). Therefore, the quality of service (QoS) plays a vital role [4]. Nowadays, the *QoS* is regulated by service-level agreements (SLAs). These are contracts between client and providers that express the price for a service, the *QoS* levels required during the service provisioning and the penalties associated with the *SLA* violations. Hence, service providers must design these contracts very carefully to maintain user confidence and avoid revenue loss [5]. The quality of service and user satisfaction have a significant relation: A lower level of *QoS* due to an *SLA* violation leads to a decrease in user satisfaction. The main challenge for a service provider is to determine the best trade-off between profit and customer satisfaction. This work is aimed at assisting service providers to design the optimal *SLA* contract regarding cost, performance and availability to maintain user satisfaction and the quality of service at the desired levels.

There is a rapidly growing trend in developing *SaaS* and real-time applications that open the door to new challenges in cloud-based hosting. The most crucial are to ensure

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

high availability (*HA*) and a reasonable response time (*RT*). To address these challenges, the central objective of this work is to design a decision model that minimizes the cost of the system and maximizes the *SLA* guarantees (based on the availability and response time of the system).

Availability is defined as the probability of an adequate *QoS* [6]. Moreover, availability is the long-term fraction of the time of service actually delivered. Short outages can usually be accepted, but more prolonged interruptions or accumulated disruptions exceeding a certain threshold may not be tolerable. The IEEE defines *HA* as the ability to wake components that have failed in the system. Morrill *et al.* [7] consider that the minimum level of *HA* is around 99.7%. Other cloud-computing issues, such as variability [8], system security [9] and reliability [10] were discarded.

Inspired by the work in [11]–[13], we represent the availability of cloud service as a Markov model, where the times between failures are exponentially distributed. Furthermore, a *M/M/1/k* queue was chosen to model the performance of a Cloud offering *SaaS*. The model consists of a cloud made up of N virtual machines (VMs). Considering the number of tasks as b, from here on $k = bN$, and the queue is renamed *M/M/1/bN* [14]. This queue was chosen not only as it fits in this Cloud environment properly but also because it allows the cloud's administrator to compute the response time quickly and reliably.

Optimizing the performance of data centers is prominent in the literature on cloud computing, see [12], [15], [16]. In this paper, the proposal is focused on designing a non-linear multi-criteria model (NLP) that minimizes the cost of the system while preserving *SLA* guarantees (based on the performance and availability). It is known that *NLP* problems are commonly harder to resolve to require more time and computational power. Nevertheless, in this case, a non-linear function represents the real behavior of the cloud service more accurately and the experiments suggest that it can be solved quickly.

There are different metrics for evaluating the performance of computer applications, such as the response time or throughput. The mean response time is perhaps the most significant performance metric in a cloud-computing context [17] dealing with *HPC* applications and real-time services, and so, this was the performance parameter chosen for this work. Hence, it is a negotiated *SLA* required to guarantee the *QoS* considered in our formulation. Moreover, the model proposed is also focused on guaranteeing a negotiated level of *QoS* regarding availability. Thus, another *SLA* is required to guarantee the aforementioned specific availability rate.

To address *QoS* requeriments regarding availability, performance (response time) and cost, this paper makes the following contributions:

(i) Modeling the cloud architecture using a queuing network model.
(ii) Modeling the availability using a Markov model.
(iii) Proposing a *SLA* decision model (CART) capable of optimizing and analyzing the cost and quality impact in the hosting of cloud-based applications (i.e. *SaaS*). This model keeps user satisfaction and quality of service (availability and performance) at a negotiated SLA.
(iv) Making an accessible CART model through a cloud-based service (AOS, Application Optimization Service) to assist cloud designers in the evaluation of their models. CART is available for everyone in AOS,[1] by clicking on "New session", and then "Cloud/CART".

II. LITERATURE REVIEW

Research related to *SLA*-based cost optimization and customer satisfaction is becoming an important branch in the Cloud computing area. Most of the works are focused on models oriented towards *IaaS* providers. For example, García *et al.* [18] propose an *SLA*-driven architecture for automatic provision, scheduling, allocation and dynamic management of cloud resources.

Nevertheless, one of the main challenges of cloud service providers is to ensure the quality of service by guaranteeing the *SLA* contract. Serrano *et al.* [19] propose a method that combines *QoS* with *SLA* in clouds aiming at facing challenges such as better performance, dependability or cost reduction of online cloud services. The paper shows advantages from providing *IaaS* and *PaaS*. Moreover, Hussain *et al.* [20] describe this situation for small-medium enterprises (SMEs). This work presents an exhaustive review of the current state of the art and highlights the main gaps in the research. They claim that a lack of a viable *SLA* management framework could lead to service interruption and contract violation penalties. Furthermore, service interruption and contract violation penalties affect customer satisfaction and cloud service trustworthiness directly.

In this context, availability, cost, reliability, dependability, performance and other cloud metrics are essential quality factors to design cloud-based services. The literature abounds with different modeling approaches that mix a subset of these metrics, see [21]. Wu *et al.* [22] present an optimization model to minimize cost and improve customer satisfaction level considering *SLA* violations and response time. Kouki and Ledoux [23] propose an analytical model to predict cloud service performance regarding the cost and the dependability of the service.

High Availability in cloud computing services is one of the most crucial quality metrics [24]. *HA* for cloud services is essential for maintaining customer confidence and preventing revenue losses due to *SLA* violation penalties [25]. Snyder *et al.* [26] claim that about $285 million have been lost yearly due to cloud service failures.

Although many efforts have been dedicated to analyzing the availability of cloud (or web) hosts using measurement-based techniques [27], [28], less emphasis has been placed on modeling web service availability taking into account the impact of server node failures and performance degradation [10]. The modeling of the availability of

[1] http://stormy02.udl.cat/aos

fault-tolerant cloud computing systems using a Markov model to fulfill the client request was proposed in [29]. This work discusses various heterogeneous availability models for a few failure structures and designs and compares different recovery techniques.

Meaningful information generated by scheduling problems should be merged with job availability in clouds. This way, we detected an increasing number of research papers studying scheduling problems that are subject to machine or job availability constraints [12], [15], [16]. In this context, machines or jobs may be unavailable for distinct time intervals. In [12], job behavior regarding unavailability constraints was analyzed. An integer linear programming (*ILP*) algorithm was proposed for scheduling problems that occur when the weighted number of late jobs that are subject to deterministic machine availability constraints must be minimized.

Frequently used methods of modeling computer service performance subjected to such *QoS* metrics as response time, throughput and network utilization have been extensively studied in [30]–[35]. Xiong and Perros [30] obtained the response time distribution of a cloud system modeled on a classic *M/M/m*, assuming an exponential density function for the inter-arrival and service times. Yang *et al.* [32] obtained the response time distribution for a cloud with an *M/M/m/m+r* system model. Both inter-arrival and service distribution times were assumed to be exponential, and the system had a finite number of *m+r* size buffers. The complexity of other queues (*G/M/m*, *M/G/m*, *G/G/m*) comes from the impossibility of obtaining a closed formula to represent the probability distributions of the response or waiting times of customers in the queue, and therefore approximate models must be found [36]. Several authors propose analytical models based on queuing theory to estimate the performance of a heterogeneous data center accurately, for instance [37] and [38].

Recent research has indicated that the speed of service, response time and service cost are not the only crucial factors in a cloud system [39]. The growth in the literature is particularly concentrated on power consumption, with some works related to the possibility of dynamically consolidating traffic flows dynamically on as few links as possible and turning off unused links and switches [40], [41]. Other authors recommend shutting down the spare nodes to make the system greener and more efficient concerning energy consumption [42], [43]. Another option is to select the right data center considering energy efficiency, *QoS* and *SLA* [44]. This way, Rossi *et al.* [45] propose an eco-orchestration approach to balance the energy-efficiency of a data center with a smaller impact on performance. Despite the importance of energy-efficient systems in cloud computing, this work only evaluates the *QoS* based on guaranteeing the *SLA* concerning availability, response time and cost.

III. CART MODEL

In this section, the cloud architecture is presented. Next, an availability model is proposed. A novel and attractive feature of this approach is the use of non-linear optimization techniques to take advantage of the strengths of both well-studied models in the current state of the art (see the literature review, Section II).

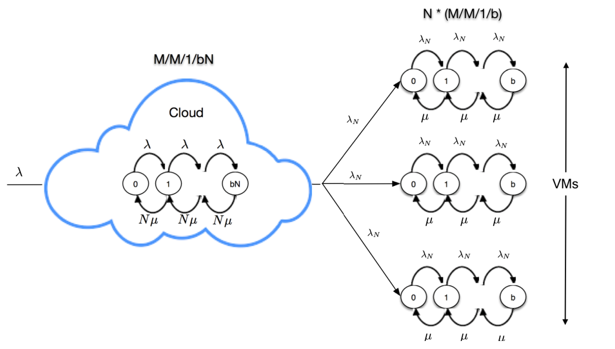

FIGURE 1. Cloud architecture model. One M/M/1/bN or N M/M/1/b queues. This figure represents the behavior of a web cloud made up of *N* slaves governed by a server.

A. CLOUD ARCHITECTURE MODEL

The performance of the cloud system proposed in this work is modeled by finite customers and a single server queue (M/M/1/bN), where N and b respectively represent the number of virtual machines and the capacity of each VM in the cloud. The cloud is made up of N virtual machines. Accordingly, the performance of a VM in the cloud is modeled by a queue (M/M/1/b). Figure 1 depicts the cloud system architecture. This figure shows that the entire cloud can be modeled using an M/M/1/bN queue. This can be broken down into N M/M/1/b queues.

The input traffic is modeled by a Poisson process with rate λ tasks/s (tasks per second). The authors chose a Poisson process for simplicity and for fitting the behavior of the web traffic properly. Since the cloud system has N available VMs, there will be N independent Poisson arrivals each with a $\lambda_N = \frac{\lambda}{N}$ rate. It is modeled like this to obtain a balanced system where each virtual resource has the same capacity. It means that each virtual resource receives a proportional amount of the workload. Moreover, each VM in the system has a service rate of μ tasks/s. Provided that the queuing model on this occasion is an M/M/1/bN, the response time (RT) can be obtained as follows:

$$RT = \frac{\hat{N}}{\lambda(1 - p_{bN})} \tag{1}$$

where $\hat{N}$ represents the mean number of VMs in the system and is defined as:

$$\hat{N} = \rho * p_0 * \sum_{k=1}^{bN} \rho^k \tag{2}$$

and ρ represents the utilization factor and is defined as:

$$\rho = \frac{\lambda}{N\mu} \tag{3}$$

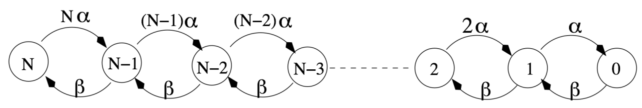

FIGURE 2. Markov model for availability.

Besides, p_i is the steady probability of having i tasks in this queue and is computed as:

$$p_i = \begin{cases} \dfrac{\rho^i(1-\rho)}{1-\rho^{bN+1}}, & \text{if } \lambda \neq N\mu \\ \dfrac{1}{bN+1}, & \text{otherwise} \end{cases}$$

If $i = bN$, the queue is full. This means that the system has no more free resources to execute new incoming tasks. Therefore, until the finalization of some tasks currently being processed, the system rejects all new incoming tasks.

B. AVAILABILITY

In this section, the availability model, inspired by the work presented in [10], is presented. The times between virtual machine failures are exponentially distributed at the rate α. We suppose that failures are not detected immediately. When failing, the system is reconfigured by restarting the virtual machine. After a virtual machine failure, the sum of the detection failure, system reconfiguration, restoration and reintegration times is assumed to be exponentially distributed at rate β.

Figure 2 shows the queuing model that describes this process. The N states of the Markov model represent the availability of the N virtual machines making up the cloud system used in the previous section.

The unavailability of the system can be defined as:

$$unavailability = \sum_{n=1}^{N} P_n L(n) + P_0, \tag{4}$$

where n represents the number of available VMs in the system. The state of the system represents the number of available VMs at any given moment. Thus, the steady-state probabilities, defined in eq. 5, represent the probability of having n available VMs at a specific time.

$$P_n = \frac{N!}{n!}\left(\frac{\alpha}{\beta}\right)^{N-n} P_N \tag{5}$$

Specifically, P_N and P_0 are defined in Eqs. 6 and 7 respectively.

$$P_N = \left[\sum_{i=0}^{N} \frac{N!}{(N-i)!}\left(\frac{\alpha}{\beta}\right)^i\right]^{-1} \tag{6}$$

$$P_0 = N!\left(\frac{\alpha}{\beta}\right)^N P_N \tag{7}$$

$L(n)$, defined in eq. 8 represents the loss probability due to a lack of capacity at VM n.

$$L(n) = np_b, \tag{8}$$

where p_b as defined in eq. 9 is the probability of loss due to a buffer overflow.

$$p_b = \begin{cases} \dfrac{\rho^b(1-\rho)}{1-\rho^{b+1}}, & \text{if } \lambda \neq \mu \\ \dfrac{1}{b+1}, & \text{otherwise} \end{cases} \tag{9}$$

Considering that the probability must be inside the interval [0-1], the availability can be defined as 1 minus the unavailability rate. Thus:

$$availability = 1 - unavailability \tag{10}$$

C. CART MODEL

In this section, an optimization model is presented to provide the ideal number of virtual machines N required to guarantee a given level of availability (passed as an argument by some clients), while minimizing the response time and also the cost of the system. Let us consider the system in the cloud. The goal of the model is to find a balanced solution between cost and response time to guarantee a certain level of availability. In this context, the model finds the best trade-off to ensure the availability and performance of the *QoS*. This is achieved by mixing two mathematical approaches, queuing theory and nonlinear programming.

To that end, an objective function (*OF*) and its constraints are presented. The *OF* corresponds to the response time defined in eq. 1. In this case, we are interested in finding the number of virtual machines in the system that minimizes eq. 1. The requirements of this model are to set two main parameters that act as constraints in the model presented in this section. The first one is *SLA_Availability*, which represents the availability agreed with the customers. This way, the model can guarantee a negotiated level of availability for the *QoS*. The second one is the *SLA_ResponseTime*, which represents an upper-bound for the response time that the administrator of the system considers quick enough to serve the jobs.Moreover, this is also a crucial factor that the *SaaS* provider must agree with the customers in the *SLA* contract. In this work, the metric used to estimate this bound is efficiency. The response time used in this model is the minimum time that does not improve the efficiency significantly, taking into account a mean number of users. The efficiency can be evaluated considering the speed-up in executing a task in a single node and the time to execute a task when more processors are added to the system, always taking into consideration a mean number of user petitions. There is overwhelming evidence of the relationship between the cost and minimization of the response time. Thus, this constraint is needed to make a trade-off between the cost and the response time.

The system is used to design the central virtual architecture we need to maintain the availability levels while serving users with high efficiency. It is true that this architecture will not always be static. The cloud is elastic and capable of adapting to changes in the workload. Thus, a cloud-based platform must be capable of adjusting to the workload. If the workload decreases, some virtual resources can be stopped. This amount can be obtained by recalculating the model. The same can be applied if the workload increases. So, the model can be recalculated when significant changes in the workload are detected. Furthermore, the results of the model are the minimum resources the system needs in a typical situation. However, the model can be recalculated to fit changes in the workload.

This function is formally defined by the following non-linear programming model:

$$OF : min \; [RT] \quad (11)$$

$$s.t.: \; Availability \geq SLA\_Availability \quad (12)$$

$$ResponseTime \leq SLA\_ResponseTime \quad (13)$$

(11) is the *OF* to be minimized. Note that the resolution of such an equation is a non-linear problem. (12) is the *SLA* constraint regarding availability (defined in 10) the system must guarantee. (13) is another constraint for ensuring the agreed performance regarding response time. Prior to designing the *SLA* with the customers, cloud providers can restrict the unnecessary use of resources (computing nodes, cores, etc.) to satisfy users and keep the *QoS* at the desired level. Given the constants α, β, λ *and* μ, the solution that minimizes *OF* will obtain the value of the variable *N*, representing the number of virtual machines. (13) is used to obtain the most efficient *N* (i.e. minimum N), that is, an optimal *OF* but taking into account the number of resources.

The *N* obtained when applying the model will be the minimum number of virtual resources to ensure the *QoS* concerning the availability and performance required by the client.

IV. RESULTS

In this section, we present the results that corroborate the advantages of the model proposed. First of all, we describe the test-bed used in this study. Then, the model is evaluated to ensure the correctness of the results and highlight the influence of the availability and response time of the *SLA* on the design of cloud services. Finally, a real cloud service is deployed and monitored to show how the model proposed can simulate real systems and also to give useful guidelines for designing cloud services.

A. TEST BED

A small range of cloud architectures was considered and analyzed to show the good behavior of the model presented in section III-C. The main characteristics of each architectural-cloud design (Clouds 1,2 and 3) are described in Table 1. λ is the average arrival rate of tasks. μ is the task service rate of the system. α is used to obtain $1/\alpha$, which represents the mean time for failure detection. β is used to calculate $1/\beta$, which represents the mean time for restarting a virtual machine; b represents the capacity of each virtual machine.

Table 2 describes the initial decision to optimize the system. As explained in section III-C, *SLA_ResponseTime* represents the minimum response time that the system must ensure and *SLA_Availability* represents the rate of availability that the system must guarantee.

TABLE 1. Cloud architectural characteristics.

Name	μ	α	β	b	λ
Cloud 1	5 tasks/s	20 s	200 s	20 tasks	20 tasks/s
Cloud 2	5 tasks/s	20 s	200 s	20 tasks	40 tasks/s
Cloud 3	15 tasks/s	20 s	200 s	20 tasks	800 tasks/s

TABLE 2. Numerical values of the decision parameters.

Name	*SLA_Availability*	*SLA_ResponseTime*
P1	0.7	0.01
P2	0.8	0.001
P3	0.85	0.08

In an attempt to test the veracity of the model proposed in this paper, the authors selected cloud characteristics randomly. In spite of this, the decision parameters displayed in Table 2 were carefully chosen to reflect different trends and enrich the discussion of the results.

The model presented and discussed in Section III-C was implemented and solved using Python [46] and the Sage 8.2 mathematical software [47].

B. MODEL EVALUATION

This section presents the results obtained by the model to study the impact of the minimization on the response time that guarantees a negotiated *SLA* regarding availability when increasing the number of VMs.

The principal results obtained from the test-bed alternatives (Cloud 1, 2 and 3) and the three combinations of the decision parameters *SLA_Availability* and *SLA_ResponseTime* (P1, P2 and P3) are presented in this section. These show the relationship between the number of virtual machines, the response time and availability.

Figures 3 and 4 depict the behavior of the availability and response time in all the clouds studied. Thus, the results show the impact on availability and response time when the cloud service is offered with a certain number of virtual machines (N).

Those figures highlight the influence of the number of virtual machines on the minimization of the *RT*, guaranteeing the availability and response time of the *SLA*. This way, a range between 1 and 50 virtual machines was used to evaluate Cloud 1, Cloud 2 and Cloud 3 architectures on the test-bed. The results show the different trends in the metrics analyzed.

These figures also show the minimum number of virtual machines required to ensure the level of the response time and availability of the *SLA* agreed with the customer, represented by marker-lines (−, P1), (+, P2) and (x, P3).

Figure 3 presents a decreasing trend between response time and the number of virtual machines. Thus, the more VMs the system has, the quicker it will reply. On the contrary,

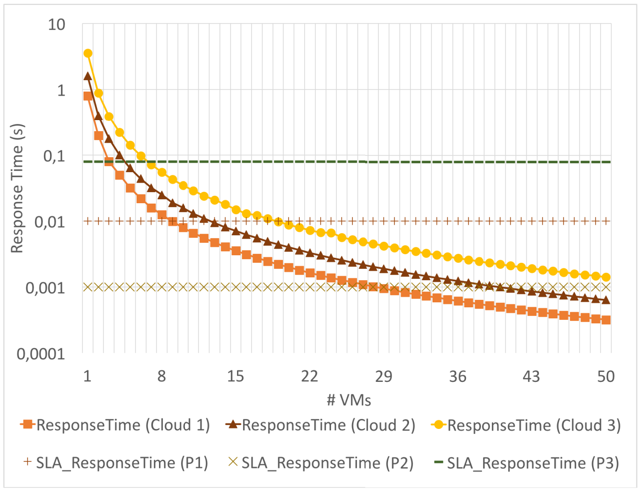

FIGURE 3. **Response time.**

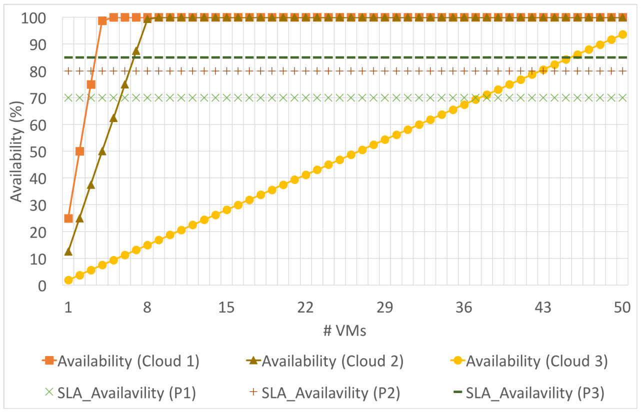

FIGURE 4. **Availability.**

an incremental trend between the availability and number of virtual machines can be appreciated in Figure 4. Therefore, the more virtual machines the system has, the more available it will be. Cloud 3 and *SLA_ResponseTime*(*P*3) show a case in which there is not enough with 50 VMs to ensure a reasonable *Response Time*. In this situation, the cloud designer must invest in more than 50 VMs to satisfy their consumers. However, this bound on the response time can be incremented to reduce the overall cost. The final decision of whether to increase either this bound or the number of virtual machines will only depend on the trade-off between customer satisfaction and the system cost.

The minimum response time is found with 50 VMs in all the clouds. Note that after a certain number of VMs, the response time barely decreases, and ceases to be an essential metric to consider. The system cost increases with the number of VMs and the type of the contract agreed. Thus, a multi-objective minimization objective is justified for designing cloud services.

For Cloud 1 and 2, the principal influencing factor in economic terms (investing in VMs) is the response time, since the *SLA* is easy to achieve regarding availability in this context (7 VMs to guarantee SLA_Availability as much for P2, see Figure 4). However, in Cloud 3, availability is the most critical aspect. Thus, Cloud 1 needs 3 VMs to ensure *SLA_Availability* (P1) and 4 for *SLA_Availability* (P2) and *SLA_Availability* (P3), whereas Cloud 3 needs 37, 45 and 47 VMs respectively. The system cost increases with the number of VMs and the type of the contract agreed. This assumption again justifies the need for a multi-objective minimization model for designing cloud services.

These results provide confirmatory evidence about the differences between designing clouds to ensure the response time and availability of the *SLA*.

C. CASE STUDY

In this section, a practical case study was designed to show how the proposed model can be applied in a real cloud environment. The cloud architecture analyzed offers software-as-a-service (*SaaS*). The application tested is a traditional cloud service. The client based on a web application performs tasks using a back-end to manage virtual machines in the cloud. The main steps are to deploy the VM, send and execute the job, send the results to the client and destroy the virtual machine. Furthermore, there is a queue to store the undone tasks. The fundamental objective is to show how the model proposed reflects reality and is helpful for *SaaS* providers to deploy the cloud architecture.

The experimental results were obtained using the Apache JMeter tool [48]. This can be used to simulate loads in a web service to test its strength or analyze overall performance by sending scheduled hypertext transfer protocol (*HTTP*) requests. The results presented in this section show the response time and availability of the system described above when monitoring the system with 1, 2 and 3 virtual machines contracted in the Amazon Web Services (*AWS*) to serve the incoming petitions. This study draws on the monitorization conducted by the JMeter tool for 1 minute and only contracted resources were used.

The average input and service task rate used in the study were $\lambda = 15$ $tasks/s$ and $\mu = 5$ $tasks/s$. Therefore, the maximum capacity of each VM was restricted to five tasks (this is $b = 5$). The virtual machines (VMs) used were *t1.micro*. The main features of this type of VM instance are computational power provided by 1.7 GB of RAM and 1 vCPU (Virtual CPU). These are aimed at general-purpose applications.

First of all, the simulation tool was used to verify the behavior of the cloud service. With the parameters described above, the cloud system solved by simulation is depicted in Figure 5. This figure shows that with contracting 3 VMs, the response time and availability of the *SLA* are satisfied.

Figures 6 and 7 show the number of incoming tasks per second (demand) generated by JMeter (right axis) and the response time or availability obtained by each cloud configuration (1, 2 or 3 VMs). Moreover, these pictures contain the desired levels (agreed with the customers) of response time and availability.

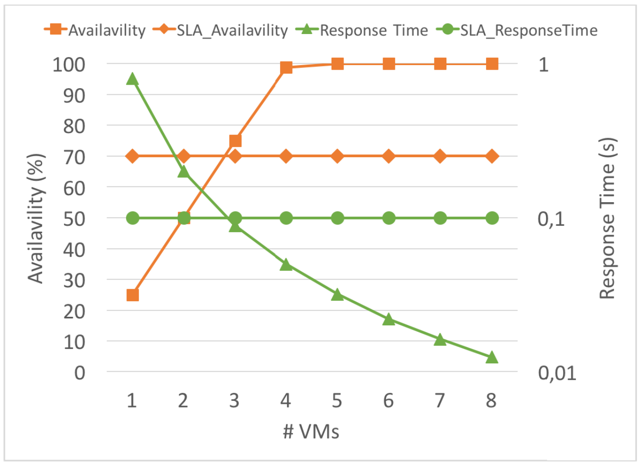

FIGURE 5. Case Study. Model results.

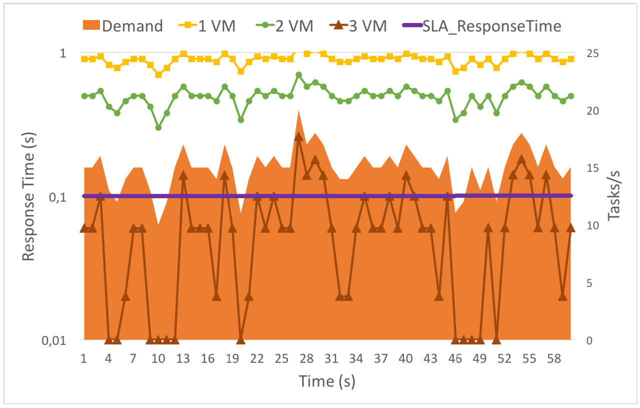

FIGURE 6. Case Study. Response time obtained with JMeter.

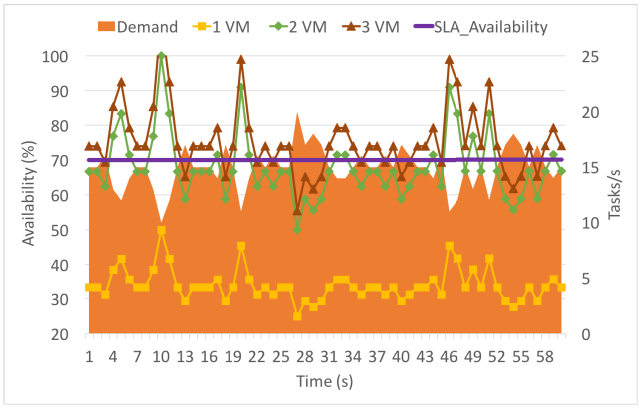

FIGURE 7. Case Study. *SLA* availability obtained with JMeter.

A closer look at the data indicates that the model solved by simulation (Figure 5) generates similar information (response time and availability) as the output obtained from monitoring the real system. Note that the percentage of availability increases while the response time decreases when more virtual machines are contracted.

Nevertheless, in the data obtained from monitoring the real system, moments in time were found when the response time and availability of the *SLA* are not satisfied. Therefore, the response time and availability exceed the SLA_ResponeTime and SLA_Availavility respectively. In all these situations, we can ensure response time and availability by deploying pay-per-use instances but with a higher cost, as explained above.

Thus, using the model proposed, we can optimize the computational resources deployed. These demand peaks are produced by the high variability of the input tasks generated by JMeter, which attempts to emulate the real load of cloud data centers and websites. Considering the results of the model, this service will contract 3 virtual machines to guarantee *SLA*. Despite this, pay-per-use instances will be required when demand increases. In this study, pay-per-use instances will be required around seconds 13, 18, 27-30, 40, 53-57 in the response time and 13, 19, 17-31, 40-41 and 52-56 in the availability.

Regarding the cost, for certain cloud providers like Amazon, the user can invest in pay-per-use or reserved capacity. With reserved resources, up to 74% can be saved compared with equivalent on-demand capacity. For example, the yearly fee of an instance *t1.micro* in the pay-per-use modality is around €175.68. Meanwhile, with a reserved capacity contract, the price is reduced to €103. In this study, the model suggests 3 VMs (reserved capacity), so we will have an annual cost of around 3 VMs*€103 = €309. However, without planning and using the pay-per-use modality, the yearly cost would be around 3VMs*€175 = €525. In this case, the savings amount to 41%. Thus, using the solution in the model proposed, it is possible to plan how many reserved capacity contracts to deploy to ensure the *SLA* and then the demand variations can be satisfied with pay-per-use resources.

This experiment shows how the model proposed represents the behavior of a real service and can be used as a guideline to evaluate and design the response time and availability in new cloud services.

V. DISCUSSION

The cloud computing literature abounds with fruitful applications for ensuring the *SLA* either through availability [11] or response time [36]. However, there are fewer applications and models in the *SaaS* that consider both metrics to ensure *QoS*. The model presented in this paper contributes to providing a decision *SLA* model for *SaaS* providers to maintain *QoS* regarding availability and performance.

The industry demands tools to minimize overall costs by offering the best *QoS* to keep the user satisfaction and avoid paying *SLA* violations. The model presented in this paper is aimed at a practical application for the design of cloud services. It provides multi-criteria features for evaluating availability, response time and cost together simultaneously.

The results presented show that sometimes the number of VMs needed to ensure availability is higher than the one required to satisfy the response time condition, and vice-versa. The relationship between the availability of an *SaaS* and client satisfaction is well known. In an *SaaS* context, the risk of losing clients due to a lack of service is very high. The same is also true for the *RT* criteria. Thus, for real businesses, it is crucial to have a multi-criteria model that ensures the satisfaction of the clients.

The results presented in Section IV provide convincing evidence about the correctness of the proposed model. It is proved that the information gathered from monitoring the real service is similar to that obtained by solving the model.

Morrill *et al.* [7] consider that the minimum level of *SLA* regarding availability is around 99.7%. With the proposed model, we provide a tool to design a cloud-based service with the best trade-off between cost and *SLA* concerning response time and availability.

Given the interest in deciding the number of contracts required for a service, cloud designers can use the model proposed to find out the minimum number of these needed to ensure a certain level of *SLA* concerning availability and response time. In the case study presented, this was contracting three reserved instances. This way, the system's credibility of good practices and service increases among the customers.

The data gathered in the results section suggest significant advantages to using the model, including the neglective cost and short time required to obtain a solution. Besides, the model can also be used to simulate different scenarios of *SLAs* regarding availability or response time. This way, cloud designers can check the differences between these configurations and check the best scenario for availability the cloud provider can offer its customers without compromising the cost.

To sum up, the model presented is capable of dealing with a considerable amount of information to make the best trade-off to keep *QoS* regarding availability, performance and cost and be used as a basis for designing the associated *SLA* contracts.

VI. CONCLUSIONS AND FUTURE WORK

The question under discussion in this study was the applicability of an optimization model capable of designing and evaluating cloud services. The results of this research provide confirmatory evidence that the combination of queuing theory models and optimization techniques can be applied efficiently in offering *SaaS*. Furthermore, the model was tested by comparing the solution of the model with the data gathered from monitoring a real system working under the same conditions. The data yielded by this study provide substantial evidence that the implementation of this technique is effortless and low-cost. Moreover, the results presented concerning availability and response time show the strength and advantages of using this technique to ensure *SLA*. This way, cloud providers can offer the users better and more reliable contracts. In the future, this model will be extended to consider another essential cloud *QoS* metric, the reliability of a cloud system. Hence, it would be fascinating to extend the proposed model for jointly evaluating the cloud service regarding availability, reliability, performance, energy savings and cost in order to design more accurate *SLA* contracts and increase user confidence and satisfaction.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generat. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X08001957

[2] B. Varghese and R. Buyya, "Next generation cloud computing: New trends and research directions," *Future Gener. Comput. Syst.*, vol. 79, pp. 849–861, Feb. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X17302224#b1

[3] P. M. Mell and T. Grance, "The NIST definition of cloud computing," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. 800-145, 2011. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

[4] A. Abdelmaboud, D. N. A. Jawawi, I. Ghani, A. Elsafi, and B. Kitchenham, "Quality of service approaches in cloud computing: A systematic mapping study," *J. Syst. Softw.*, vol. 101, pp. 159–179, Mar. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121214002830#bbib0066

[5] M. H. Ghahramani, M. Zhou, and C. T. Hon, "Toward cloud computing QoS architecture: Analysis of cloud systems and cloud services," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 1, pp. 6–18, Jan. 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7815547/

[6] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance Evaluation With Computer Science Applications*, 2nd ed. Hoboken, NJ, USA: Wiley, 2006.

[7] H. Morrill, M. Beard, and D. Clitherow, "Achieving continuous availability of IBM systems infrastructures," *IBM Syst. J.*, vol. 47, no. 4, pp. 493–503, 2008.

[8] A. Iosup, N. Yigitbasi, and D. Epema, "On the performance variability of production cloud services," in *Proc. 11th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, May 2011, pp. 104–113. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5948601/

[9] J. Varia. (2010). *Architecting for the Cloud: Best Practices*. [Online]. Available: http://docs.huihoo.com/amazon/aws/whitepapers/AWS-Cloud-Best-Practices-January-2011.pdf

[10] M. Martinello, M. Kaâniche, and K. Kanoun, "Web service availability—Impact of error recovery and traffic model," *Rel. Eng. Syst. Saf.*, vol. 89, no. 1, pp. 6–16, Jul. 2005. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0951832004001723

[11] T. A. Nguyen, D. S. Kim, and J. S. Park, "Availability modeling and analysis of a data center for disaster tolerance," *Future Gener. Comput. Syst.*, vol. 56, pp. 27–50, Mar. 2016. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167739X15002824

[12] B. Detienne, "A mixed integer linear programming approach to minimize the number of late jobs with and without machine availability constraints," *Eur. J. Oper. Res.*, vol. 235, no. 3, pp. 540–552, Jun. 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0377221713008758

[13] D. A. Menasce, "Performance and availability of Internet data centers," *IEEE Internet Comput.*, vol. 8, no. 3, pp. 94–96, May 2004. [Online]. Available: http://ieeexplore.ieee.org/document/1297280/

[14] L. Kleinrock, *Queueing Systems: Computer Applications*, vol. 2 and 66. Hoboken, NJ, USA: Wiley, 1976, [Online]. Available: http://www.biruni.tn/gw_2009_4_3/thumbs/contents-table/38/TM.383052.pdf

[15] N. Hashemian, C. Diallo, and B. Vizvári, "Makespan minimization for parallel machines scheduling with multiple availability constraints," *Ann. Oper. Res.*, vol. 213, no. 1, pp. 173–186, Feb. 2014. [Online]. Available: http://link.springer.com/10.1007/s10479-012-1059-8

[16] W. Kubiak, J. Błazewicz, P. Formanowicz, J. Breit, and G. Schmidt, "Two-machine flow shops with limited machine availability," *Eur. J. Oper. Res.*, vol. 136, no. 3, pp. 528–540, Feb. 2002. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0377221701000832

[17] R. Aversa, B. Di Martino, M. Rak, S. Venticinque, and U. Villano, "Performance prediction for HPC on clouds," *Cloud Comput., Princ. Paradigms*, pp. 437–456, 2011. doi: 10.1002/9780470940105.ch17.

[18] A. G. García, I. B. Espert, and V. H. García, "SLA-driven dynamic cloud resource management," *Future Gener. Comput. Syst.*, vol. 31, pp. 1–11, Feb. 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X1300215X

[19] D. Serrano *et al.*, "SLA guarantees for cloud services," *Future Gener. Comput. Syst.*, vol. 54, pp. 233–246, Jan. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X15000801

[20] W. Hussain, F. K. Hussain, O. K. Hussain, E. Damiani, and E. Chang, "Formulating and managing viable SLAs in cloud computing from a small to medium service provider's viewpoint: A state-of-the-art review," *Inf. Syst.*, vol. 71, pp. 240–259, Nov. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306437917302697

[21] D. Ardagna, G. Casale, M. Ciavotta, J. F. Pérez, and W. Wang, "Quality-of-service in cloud computing: Modeling techniques and their applications," *J. Internet Services Appl.*, vol. 5, no. 1, p. 11, Dec. 2014. [Online]. Available: http://jisajournal.springeropen.com/articles/10.1186/s13174-014-0011-3

[22] L. Wu, S. K. Garg, S. Versteeg, and R. Buyya, "SLA-based resource provisioning for hosted software-as-a-service applications in cloud computing environments," *IEEE Trans. Services Comput.*, vol. 7, no. 3, pp. 465–485, Jul. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6654162/

[23] Y. Kouki and T. Ledoux, "SLA-driven capacity planning for Cloud applications," in *Proc. IEEE 4th Int. Conf. Cloud Comput. Technol. Sci.*, Dec. 2012, pp. 135–140. [Online]. Available: http://ieeexplore.ieee.org/document/6427519/

[24] M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh, "Reliability and high availability in cloud computing environments: A reference roadmap," *Hum.-Centric Comput. Inf. Sci.*, vol. 8, no. 1, p. 20, Dec. 2018. [Online]. Available: https://hcis-journal.springeropen.com/articles/10.1186/s13673-018-0143-8

[25] J. Dantas, R. Matos, J. Araujo, and P. Maciel, "Eucalyptus-based private clouds: Availability modeling and comparison to the cost of a public cloud," *Computing*, vol. 97, no. 11, pp. 1121–1140, Nov. 2015. [Online]. Available: http://link.springer.com/10.1007/s00607-015-0447-8

[26] B. Snyder, J. Ringenberg, R. Green, V. Devabhaktuni, and M. Alam, "Evaluation and design of highly reliable and highly utilized cloud computing systems," *J. Cloud Comput.*, vol. 4, no. 1, p. 11, Dec. 2015. [Online]. Available: http://www.journalofcloudcomputing.com/content/4/1/11

[27] D. Oppenheimer and D. A. Patterson, "Architecture and dependability of large-scale Internet services," *IEEE Internet Comput.*, vol. 6, no. 5, pp. 41–49, Sep. 2002. [Online]. Available: http://ieeexplore.ieee.org/document/1036037/

[28] M. Kalyanakrishnan, R. K. Iyer, and J. U. Patel, "Reliability of Internet hosts: A case study from the end user's perspective," *Comput. Netw.*, vol. 31, nos. 1–2, pp. 47–57, Jan. 1999. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0169755298002293

[29] D. Mani and A. Mahendran, "Availability modelling of fault tolerant cloud computing system," *Int. J. Intell. Eng. Syst.*, vol. 10, no. 1, pp. 1–12, 2017. [Online]. Available: http://www.inass.org/share/2017022817.pdf

[30] K. Xiong and H. G. Perros, "Service performance and analysis in cloud computing," in *Proc. SERVICES-I*, Jul. 2009, pp. 693–700. [Online]. Available: http://ieeexplore.ieee.org/document/5190711/

[31] L. P. Slothouber, "A model of Web server performance," in *Proc. 5th Int. World Wide Web*, 1996, pp. 571–576. [Online]. Available: http://www.oocities.org/webserverperformance/modelpaper.html

[32] B. Yang, F. Tan, Y.-S. Dai, and S. Guo, "Performance evaluation of cloud service considering fault recovery," in *Cloud Computing*. Berlin, Germany: Springer, 2009, pp. 571–576. [Online]. Available: http://link.springer.com/content/pdf/10.1007/978-3-642-10665-1.pdf#page=590

[33] B. N. W. Ma and J. W. Mark, "Approximation of the mean queue length of an *M/G/c* queueing system," *Oper. Res.*, vol. 43, no. 1, pp. 158–165, 1995. doi: 10.1287/opre.43.1.158.

[34] H. Karlapudi and J. Martin, "Web application performance prediction," Ph.D. dissertation, Dept. Comput. Sci., Clemson Univ., Clemson, SC, USA, 2004.

[35] R. D. Van Der Mei and H. B. Meeuwissen, "Modelling end-to-end quality-of-service for transaction-based services in multi-domain environments," in *Proc. IEEE Int. Conf. Web Services*, Feb. 2006, pp. 453–462.

[36] J. Vilaplana, F. Solsona, I. Teixidó, J. Mateo, F. Abella, and J. Rius, "A queuing theory model for cloud computing," *J. Supercomput.*, vol. 69, no. 1, pp. 492–507, 2014. [Online]. Available: http://link.springer.com/article/10.1007/s11227-014-1177-y

[37] W.-H. Bai, J.-Q. Xi, J.-X. Zhu, and S.-W. Huang, "Performance analysis of heterogeneous data centers in cloud computing using a complex queuing model," *Math. Problems Eng.*, vol. 2015, Jun. 2015, Art. no. 980945.

[38] S. E. Kafhali and K. Salah, "Stochastic modelling and analysis of cloud computing data center," in *Proc. 20th Conf. Innov. Clouds, Internet Netw. (ICIN)*, Mar. 2017, pp. 122–126.

[39] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based IoT applications," *J. Netw. Comput. Appl.*, vol. 89, pp. 96–108, Jul. 2017. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1084804517301108

[40] B. Heller *et al.*, "ElasticTree: Saving energy in data center networks," in *Proc. 7th USENIX Symp. Netw. Syst. Design Implement.*, 2010, pp. 249–264. [Online]. Available: https://www.usenix.org/legacy/event/nsdi10/. doi: 10.1.1.174.3815.

[41] Z. Guo, S. Hui, Y. Xu, and H. J. Chao, "Dynamic flow scheduling for Power-efficient Data Center Networks," in *Proc. IEEE/ACM 24th Int. Symp. Quality Service (IWQoS)*, Jun. 2016, pp. 1–10. [Online]. Available: http://ieeexplore.ieee.org/document/7590399/

[42] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency Comput., Pract. Exper.*, vol. 24, no. 13, pp. 1397–1420, 2012. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/cpe.1867/full

[43] D. Kliazovich, P. Bouvry, and S. U. Khan, "GreenCloud: A packet-level simulator of energy-aware cloud computing data centers," *J. Supercomput.*, vol. 62, no. 3, pp. 1263–1283, 2012. [Online]. Available: https://link.springer.com/article/10.1007/s11227-010-0504-1

[44] B. Aldawsari, T. Baker, and D. England, "Trusted energy-efficient cloud-based services brokerage platform," *Int. J. Intell. Comput. Res.*, vol. 6, no. 4, pp. 1–10, 2015. [Online]. Available: http://infonomics-society.org/wp-content/uploads/ijicr/published-papers/volume-6-2015/Trusted-Energy-Efficient-Cloud-Based-Services-brokerage-Platform.pdf

[45] F. D. Rossi, M. G. Xavier, C. A. F. De Rose, R. N. Calheiros, and R. Buyya, "E-eco: Performance-aware energy-efficient cloud data center orchestration," *J. Netw. Comput. Appl.*, vol. 78, pp. 83–96, Jan. 2017. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1084804516302569

[46] *Python Software Foundation*. Accessed: Jun. 6, 2018. [Online]. Available: http://www.python.org

[47] *Sage Mathematics Software*. Accessed: Jun. 6, 2018. [Online]. Available: http://www.sagemath.org

[48] *Apache JMeter*. Accessed: Jun. 6, 2018. [Online]. Available: http://jmeter.apache.org

JORDI MATEO-FORNÉS received the ETIG degree and the B.Sc. and M.S. degrees in computer engineering from the Escola Politècnica Superior, Universitat de Lleida (UdL) in 2006, 2012, and 2013, respectively. He is currently pursuing the Ph.D. degree. His research interests and expertise include data science fields, such as operations research (stochastic optimization), high-performance computing (parallelization of algorithms), cloud computing, big data, and the Internet of Things (IoT), decision support systems, and Agriculture 4.0.

FRANCESC SOLSONA-TEHÀS received the B.S., M.S., and Ph.D. degrees in computer science from the Universitat Autònoma de Barcelona, Spain, in 1991, 1994, and 2002, respectively. He is currently a Full Professor with the Department of Computer Science, University of Lleida, Spain. His research interests include parallelization of algorithms, coscheduling, cluster computing, multicluster computing, P2P computing, cloud computing, desktop grid computing, scheduling, optimization, multi-stage linear and stochastic programming, reconstruction of metabolic pathways, bioinformatics, ecosystems simulation, mHealth, big data, and the Internet of Things (IoT).

JORDI VILAPLANA-MAYORAL received the Ph.D. degree in computer science. He is currently a Professor with the University of Lleida and also a member of the Distributed Computing Group. He is involved in multiple interdisciplinary projects alongside clinicians, psychologists, statisticians, and mathematicians. He uses telemedicine techniques, web-based applications, smartphone applications, image recognition, data entry, big data storage, processing and visualization, machine learning algorithms, and deep learning techniques. His research interests include cloud computing, eHealth and mHealth, big data, and machine learning applied to the health field.

IVAN TEIXIDÓ-TORRELLES received the ETIS degree, the B.S. degree in computer engineering, and the M.S. in applied science to engineering from the Escola Politécnica Superior, Universitat de Lleida (UdL), Lleida, in 2007, 2009, and 2011, respectively. He is currently a Research Support Staff with UdL. His research interests include cloud computing, big data analysis, machine learning, and parallelization of scientific and technological applications.

JOSEP RIUS-TORRENTÓ received the B.S. and M.S. degrees in computer science from the Universitat de Lleida, Spain, in 2006 and 2008, respectively, the B.S. and M.S. degrees in business administration, in 2009 and 2011, respectively, and the Ph.D. degree in computer science from the Universitat de Lleida, in 2012, where he is currently an Associate Professor with the Department of Business Administration. His research interests include parallel and distributed computing, cloud infrastructures, big data, and data analysis.