



Evaluating cloud interactions with costs and SLAs

Adrián Bernal¹ · M. Emilia Cambroneró¹ · Alberto Núñez³ ·
Pablo C. Cañizares² · Valentín Valero¹

Accepted: 9 October 2021 / Published online: 22 November 2021
© The Author(s) 2021

Abstract

In this paper, we investigate how to improve the profits in cloud infrastructures by using price schemes and analyzing the user interactions with the cloud provider. For this purpose, we consider two different types of client behavior, namely *regular* and *high-priority* users. *Regular* users do not require a continuous service, and they can wait to be attended to. In contrast, *high-priority* users require a continuous service, e.g., a 24/7 service, and usually need an immediate answer to any request. A complete framework has been implemented, which includes a UML profile that allows us to define specific cloud scenarios and the automatic transformations to produce the code for the cloud simulations in the Simcan2Cloud simulator. The engine of Simcan2Cloud has also been modified by adding specific SLAs and price schemes. Finally, we present a thorough experimental study to analyze the performance results obtained from the simulations, thus making it possible to draw conclusions about how to improve the cloud profit for the cloud studied by adjusting the different parameters and resource configuration.

Keywords Cloud computing · Profit improvement · Simulation tools · SLAs

✉ Adrián Bernal
adrian.bernal@uclm.es

M. Emilia Cambroneró
memilia.cambroneró@uclm.es

Alberto Núñez
alberto.nunez@pdi.ucm.es

Pablo C. Cañizares
pablo.cerro@uam.es

Valentín Valero
valentin.valero@uclm.es
<https://www.dsi.uclm.es/personal/ValentinValero/>

¹ Albacete Research Institute of Informatics, Universidad de Castilla-La Mancha, Albacete, Spain

² Computer Science Department, Autonomous University of Madrid, Madrid, Spain

³ Software Systems and Computation Department, Complutense University of Madrid, Madrid, Spain

1 Introduction

The cloud is increasingly dominating the Information Technology (IT) markets by opening up new opportunities for new participating companies and business models. A recent report by the Synergy Research Group [1] shows that revenues for cloud providers from cloud services and infrastructures rose to more than \$150 billion in the first half of 2019, which represents an increase of a 24% since 2018 [2], largely due to the growth in the areas of Infrastructure as a Service (IaaS [3]), and Platform as a Service (PaaS [4]). This report indicates that cloud-related markets are growing at rates from 10% to 40%, and it predicts that the annual investment in cloud services will double during the following four years. In the last few years, some well-known cloud computing providers, such as Amazon Web Services (AWS [5]), Microsoft Azure [6], and Google Cloud [7] have maintained wars for IaaS, but they have preserved their position in this field, even enlarging their profits.

However, new actors such as Snowflake [8], Veeam [9], or the European Cloud Initiative [10] are playing an increasingly active role in the cloud market, so it is an essential issue for cloud providers to be more competitive and to have the ability to manage their resources in a more effective and profitable way. For this purpose, the analysis of cloud services and infrastructures, but also a close examination of how users interact with them is essential to determine their profitability. Nevertheless, it is not easy for companies to analyze the clients' behavior and draw conclusions about how to increase the utilization of their resources to improve their profits.

In this paper, we define a framework for modeling, simulating, and analyzing cloud systems to help cloud providers to maximize their profit. In essence, this framework consists of a cloud profile, an editor to configure cloud systems, and a cloud simulator. Therefore, with the help of cloud simulators, we can analyze a great variety of situations, in terms of both the cloud infrastructure and client interactions demanding cloud services.

The key component of the proposed framework, that is, the cloud profile, has been designed using UML 2.5 (Unified Modeling Language [11]), a well-accepted modeling language providing a complete set of tools for the design of systems. Specifically, we use UML component and sequence diagrams to model the relationships and the associations between the system components, the flow of actions, and the interactions among the roles in the system. In this regard, the cloud profile models the architecture and interactions in a cloud system based on Service Level Agreements (SLAs), which allows us to specify a wide spectrum of cost-related parameters, such as the cost of VMs, offers for user subscriptions, and compensation due to resource unavailability, among others [12]. Additionally, this profile can be applied to capture the main elements of both the cloud infrastructure and the client interactions to have a methodological way to model different scenarios and then launch the corresponding simulations by using a cloud simulator. We have conducted a thorough empirical study where many different cloud configurations have been created and, then, automatically translated

into the input files of a cloud simulator. Specifically, we use Model to Text (M2T) transformation techniques by applying the MOFM2T [13] (M2T transformations) standard to generate the configuration files for the Simcan2Cloud [14] simulator. For this study, we consider two types of user, *regular* users and *high-priority* users. *Regular* users do not require a continuous service, so they can wait to be attended to, while *high-priority* users pay for a continuous service, so they need an immediate answer to their requests. The analysis of these experiments would allow the cloud provider to set up the appropriate infrastructure to improve its profit at maximum.

A preliminary version of the cloud profile presented here was defined in [15], where the basic infrastructure of the cloud and the underlying interactions with the users were modeled. It is important to remark that this previous work does not consider any cost-per-use strategy. Additionally, we developed a prototype tool [14] for generating configuration files for the preliminary version of the Simcan2Cloud simulator, which is used to represent the behavior of different cloud environments. This preliminary version of the cloud profile was later extended in [16] to consider SLAs and cost per use strategies.

Our work in this paper extends the aforementioned works, that is, the proposed framework including the cloud profile, the Simcan2Cloud simulator, and the tool for automatically generating cloud environments. In the case of the profile, an SLA feature has been completely designed with two types of users, which request the cloud services and pay for their use. Furthermore, the Simcan2Cloud simulator has also been extended to provide support for these new features with a new *User generator* module, which creates fully customized users accessing the cloud. Among other novelties, this module has been extended in the present work to represent cost-related aspects, including the SLAs provided by the cloud and the behavior of the user that is processed in run-time depending on the responses received by the cloud (i.e., when the requested resources are not available or when the cloud makes an offer for renting VMs). In summary, the main contributions of this paper consist of: i) the UML profile for cloud systems with costs, the modeling tool, and the transformation; ii) the extension of the cloud simulator; and iii) the analysis of the performance evaluation in different cloud scenarios, which allow us to draw relevant conclusions about how a cloud provider can increase its profits by improving the infrastructure exploitation.

The paper is structured as follows. First, a complete description of the related work is presented in Sect. 2. The cloud simulator, named *Simcan2Cloud*, is explained in detail in Sect. 3. Section 4 describes the transformation from the UML cloud models to the Simcan2Cloud input files, and the performance evaluation of different cloud configurations is presented in Sect. 5. Section 6 describes the threats to the validity of our empirical evaluation. And finally, Sect. 7 presents the conclusions and future lines of work.

2 Related work

In recent years, cloud computing modeling, pricing schemes, and cost-benefit analysis have been a matter of great interest for the scientific community due to the considerable growth in this technology and the importance for cloud providers of being increasingly profitable.

Some authors, as García-García et al. [17], emphasize the importance of software process simulation modeling. They present a systematic review about Software Process Simulation Modelling (SPSM), where 8070 papers (from 2013 to 2019) are studied by a systematic search in 4 digital libraries. The main purpose of this work is to identify trends and directions for future research on SPSM. This review concludes that SPSM is a topic that is very much addressed by the scientific community.

In the field of cloud modeling, several UML profiles have been proposed to model different cloud aspects. For instance, Kamali et al. [18] proposed a UML profile for modeling cloud infrastructures and instances, but they do not consider the interactions of the users with the cloud or the cost-per-use of the cloud resources. Bergmayr et al. [19] presented a work considering pricing features. Specifically, they presented a language called the Cloud Application Modeling Language, or CAML for short, which is based on UML. The authors claimed that this language facilitates expressing cloud deployment topologies in UML. For this purpose, they used dedicated UML profiles. These profiles also allow the modeling of the cost of the cloud resources. However, they do not model the interactions of the users with the cloud, and SLAs are not considered either.

As regards pricing schemes, Aishwarya and Hasan [20] present a pricing scheme comparison based on certain features, such as fairness, merits, and demerits, and include a discussion of the service model and the deployment model. Chen et al. [21] have also performed a comparative study but in this case between two pricing schemes offered to cloud users by major cloud providers: the reservation-based scheme also called R-scheme, which is frequently used by Microsoft and Amazon, and the utilization-based scheme also called U-scheme, which is used by Google. Peijing et al. [22] also focused on maximizing the cloud provider profits by analyzing the pricing schemes without violating the established SLA. For this purpose, they presented a dynamic pricing model based on the concept of user-perceived value, indicating that it captures the real supply and demand relationship in the cloud service market. Another approach was used by Chi et al. [23], who presented a methodology for the dynamic pricing of the virtual machines, depending on facility and equity to deploy the requested virtual machines in the cloud. This method encourages the use of those services that can be easily deployed in the cloud, by offering cheaper prices for them. At the same time, it tries to dissuade the users from using those services that are more difficult to deploy, by using higher prices for them. Wang et al. [24] also proposed a dynamic pricing strategy. It was developed by taking into account the quality of the service perceptions of the users to assign an appropriate price. All of these works only focus on analyzing different pricing schemes offered to the users, to

maximize the cloud provider profit, but offering better prices to the users for certain products. In contrast, our work is not only focused on the study of the pricing schemes, as our main interest is to help the cloud provider to improve the use of the cloud infrastructure, to obtain a balance between the cost of this infrastructure, the current cloud user demands, and, of course, the maximum profit from it.

Concerning the cost-benefit analysis, there are some works focusing on reducing the costs for the users (user's perspective), and other works that explore how to reduce the costs for the cloud provider, maximizing its profits. Following the first approach, Mei et al. [25] proposed to include a new role in the system. This new role is called the cloud broker, which acts as an intermediary in the communication between the cloud provider and the users, and which rents virtual machines for a lower price than the cloud provider. They claimed that this cloud broker reduces the cost for the user. For this purpose, they used a heuristic method to configure the cloud broker to optimize the virtual machine pricing of the cloud broker. Fabra et al. [26] proposed a framework to generate price prediction models for Amazon EC2 Spot Instances. Prediction models are applied to generate optimal resource provisioning plans. As a result, users achieve cost savings when they use the cloud infrastructure in a real scenario. In the line of maximizing the income for the cloud provider, Manvi and Shyam [27] have analyzed how to optimize operational cloud costs by improving resource allocation. In the same way, Tang et al. [28] focused on reducing the operational costs by using the cloud resources efficiently and also reducing the system energy consumption through a proposed workflow task scheduling algorithm. Khan et al. [29] aimed at reducing energy consumption by proposing a performance efficient resource consolidation scheme for heterogeneous cloud datacenters. Herzfeldt et al. [30] have discussed different guidelines for the profitable design of the cloud service offerings. Specifically, they carried out 14 interviews with cloud provider experts, in which the relationship between value facilitation, that is, the capability to accumulate resources for future customer demands, and profitability for the cloud provider are studied. The results indicated a positive relationship between cloud service profitability and the provision of valuable services for customers. The authors claimed that cloud providers have particular difficulties to design suitable business models. This is why they were experimenting with a variety of business models to achieve a sustainable and profitable position in the cloud ecosystem.

In this work, we adopt the second perspective. Thus, we consider the cost-benefit analysis from the cloud provider perspective, but our approach is quite different from the above works. We model both the cloud infrastructure and the user interactions with the cloud provider, with the goal of obtaining the resources requested. From the UML model produced, we automatically obtain the configuration files for the Simcan2Cloud simulator, and thus we can run different simulations to analyze the cloud performance under different configurations.

3 Simcan2Cloud: a simulator for modeling cloud environments

The analysis of cloud systems is a rather complex task. There are many elements that must be considered, which involve both the physical infrastructure and the user demands. Cloud simulation has proved to be an effective technique for studying

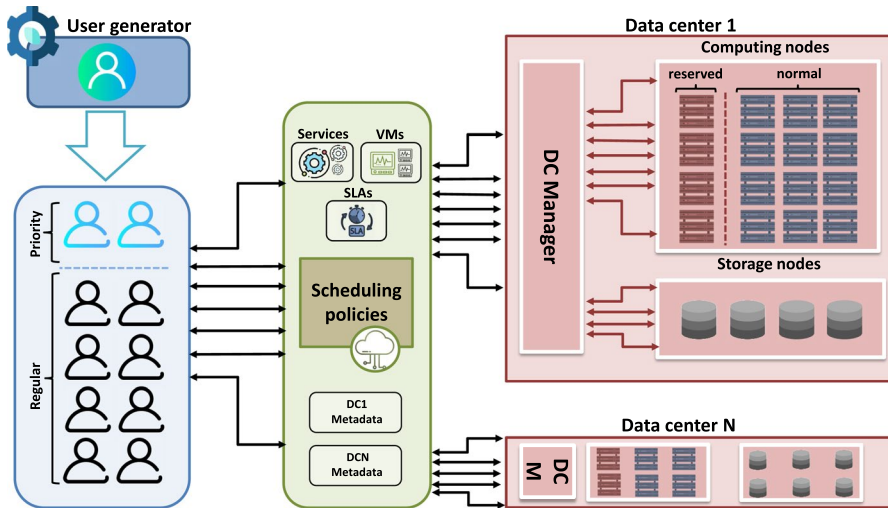


Fig. 1 General architecture of Simcan2Cloud

such systems, which requires modeling both aspects with a certain level of detail. These systems usually consist of a huge quantity of physical nodes, whose resources are provided to the users to support their computing needs. At the same time, these resources are virtualized to maximize the use of the computational elements of the cloud, which are used by a vast number of concurrent users. Hence, cloud simulation requires the use of flexible modeling techniques, and the corresponding tool support, to obtain the required level of accuracy in representing both the structure and behavior of cloud systems and in performing the quantitative analysis of these systems.

Simcan2Cloud was developed with these goals in mind. This tool is a flexible and scalable simulation platform based on SIMCAN [31, 32], which allows us to model and simulate cloud computing infrastructures. The modeling process starts with the most essential components, such as CPUs, memory, and storage systems, and then goes on to model more complex elements, such as computing and storage nodes, communication networks, and the virtualization of hardware resources.

We should mention that some parts of the simulation engine of Simcan2Cloud, such as the functionalities of sending messages along a path of modules and managing requests, have been inherited from SIMCAN. However, the underlying infrastructure and functionality of the cloud have been designed and implemented from scratch. Simcan2Cloud has been successfully used to model and simulate cloud systems in some of our previous works [14, 15]. In this new proposal, we have extended the earlier versions of this platform by providing new user profiles, SLAs, and price schemes. The main goal is to provide mechanisms that allow users to model a wide spectrum of cloud scenarios. Thus, Simcan2Cloud provides a high level of flexibility, allowing us to model and configure both the hardware and software parts of the cloud. Furthermore, a hardware repository has been implemented, making it possible to re-use the previous hardware configurations or even combine them with some

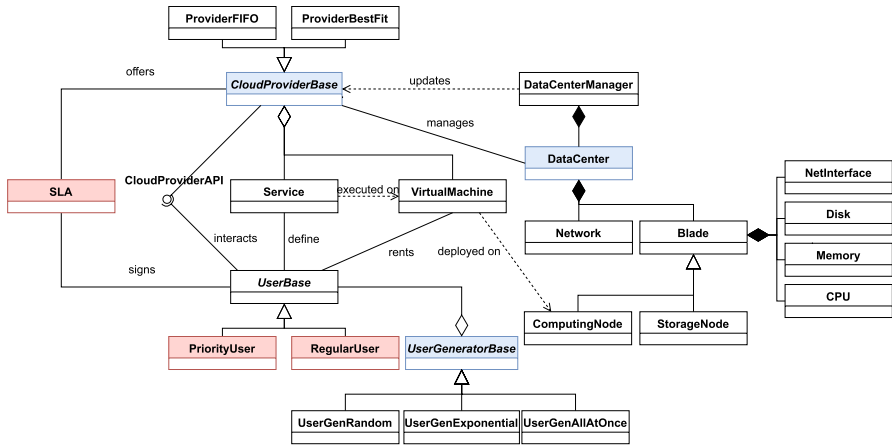


Fig. 2 Simcan2Cloud class design diagram

other elements in the repository. Regarding the users, we have included the two types of users aforementioned (*regular* and *high-priority*), new policies for managing the user requests and the SLAs.

The architecture of Simcan2Cloud is depicted in Fig. 1. To provide a high level of flexibility, the architecture of Simcan2Cloud consists of three modules that can be individually designed to generate a cloud scenario: *user generator*, *cloud provider* and *data centers*.

The *user generation* module produces users that access the cloud in the simulated scenario. In essence, each user rents one or several VMs for a specified time-frame, in which one or several services, previously requested by the user, are executed. The *cloud provider* module contains data structures and scheduling policies for handling user requests and allocating resources. Finally, the *data centers* module represents the hardware part of the cloud, i.e., the physical machines on which the VMs requested by the users are deployed.

Figure 2 shows the class diagram of Simcan2Cloud, in which only the most significant elements are depicted. Elements shown in pink are new, with respect to the previous version of the tool, while those shown in blue have been modified to include the new features in the tool. Thus, the new functionalities mentioned above have been included through the classes *SLA*, *PriorityUser* and *RegularUser*, while some existing classes, such as *CloudProviderBase*, *UserGenerator* and *DataCenter*, have been modified to fit the new features.

3.1 Generation of users

In Simcan2Cloud, the workload is represented by a collection of users accessing the cloud and executing some applications on the assigned VMs. The user behavior is implemented by using a class that inherits from *UserBase*. Thus, we have implemented a *PriorityUser* class, which represents the users who need an immediate answer to any request, and a *RegularUser* class, which represents the *regular* users.

In addition, the *UserGenerationBase* class contains the references for the services and VMs offered by the cloud, and it provides us with mechanisms to generate users and orchestrate their arrival to the cloud. For this purpose, there are three classes that produce the user accesses to the cloud, using interarrival times following an exponential distribution (*UserGenExponential*), a random function (*UserGenRandom*) or in such a way that all the user accesses are made at once, at the beginning of a simulation (*UserGenAllAtOnce*).

3.2 Cloud provider

The cloud provider is an essential part of cloud environments, since the correct behavior and the overall system performance heavily depend on its proper operation. This module is in charge of handling user requests, locating the resources requested by the users from among the available data centers, and generating an answer to these requests. More specifically, the cloud provider must accomplish four tasks: i) managing the VMs of the cloud system; ii) managing the list of jobs submitted by the users; iii) scheduling these jobs to be executed in the VM instances, and iv) defining cost policies for each VM instance type.

In this new version of Simcan2Cloud, the *CloudProvider* class has been extended with two new tasks. The first task is the supervision and management of *SLAs*, which are the contracts that establish the terms and costs of the services that the cloud provider offers, and which are signed by the users. The second new task is the handling of the requests coming from *high-priority* users, which have a rather different behavior than that of *regular* users. Specifically, *high-priority* users must have priority access to infrastructure. These users do not subscribe to the services, so they leave the cloud immediately when they cannot gain access to the resources they need and in such a case they receive compensation.

Listing 1 shows an excerpt from the API offered by the cloud provider through the *CloudProviderAPI* interface. This API provides a list of operations that can be invoked by the users to communicate with the cloud. It is important to mention that this API contains a large number of methods and, for the sake of clarity, only the most representative ones are shown in this listing.

3.3 Data center

A data center encloses the physical resources of a cloud system. These resources can be grouped into three categories: computing nodes, storage nodes and communication networks, which are represented by the classes *ComputingNode*, *StorageNode* and *Network*, respectively. A physical machine is then represented by a computational node, on which the VMs are deployed. A computational node consists of 4 subsystems: the CPU system, the memory system, the storage system, and the network system, which are represented by the classes *CPU*, *Memory*, *Disk*, and *NetInterface*, respectively.

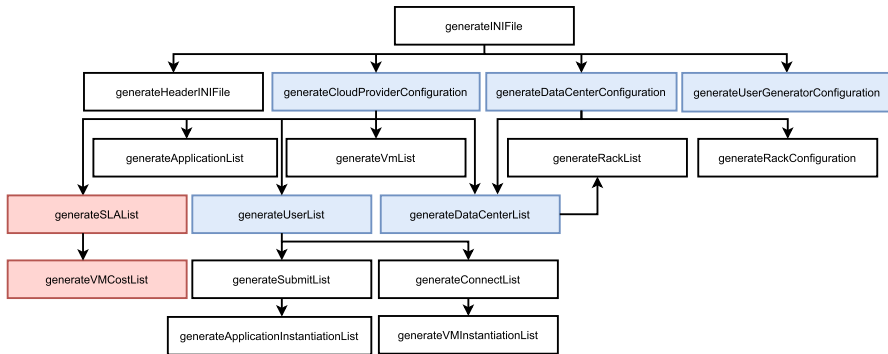


Fig. 3 Simcan2Cloud M2T .ini transformation scheme

The information about the available resources is managed by the *DataCenter-Manager* class. This class has been enriched to manage the requirements of *high-priority* users, and specifically to include the list of resources that are reserved for the exclusive use of these users.

```

1 void generateShuffledUsers ();
2 void submitService (SM_UserVM* userVm );
3 void subscribe (SM_UserVM* userVm );
4 void handleUserAppRequest (SM_UserAPP* userAPP_Rq );
5 void acceptAppRequest (SM_UserAPP* userAPP_Rq );
6 void rejectVmRequest (SM_UserVM* userVM_Rq );
7 void notifySubscription (SM_UserVM* userVM_Rq );
8 void timeoutSubscription (SM_UserVM* userVM_Rq );
9 void freeUserVms (std :: string strUsername );
10 int getTotalCoresByVmType (std :: string strVmType );

```

Listing 1 API exported by the Cloud Provider.

4 Transformation

The editor allows the user to easily produce a graphical model of the cloud system infrastructure and also of the user interactions with the cloud provider. This graphical model is then transformed into the configuration files that are required to perform the simulations. These configuration files are two files with extensions *.ini* and *.ned*, representing the configuration of each component in the system and the topology of the cloud, respectively. The transformation module of Simcan2Cloud has also been extended from the previous version [14], to include the new features related to SLAs, new types of VMs, and users.

As an illustration, Fig. 3 shows a scheme that indicates how these transformations are performed to produce the *.ini* configuration file. Two new subtransformations have been included, namely *generateSLAList* and *generateVMCostList* (in pink in the figure), which generate the list of SLAs offered by the cloud provided and the

corresponding costs for each VM, respectively. The subtransformations highlighted in blue have been modified with respect to the previous version. Thus, the *generateCloudProviderConfiguration* subtransformation has been modified to add the execution of the *generateSLAList* subtransformation. The subtransformations *generateDataCenterConfiguration* and *generateDataCenterList* have been modified to include the corresponding number of reserved machines in the code. Finally, the *generateUserGeneratorConfiguration* and *generateUserList* subtransformations have been modified to add the user types and signed SLAs.

These transformations have been implemented with Acceleo [33], an open-source code generator based on the MOFM2T (MOF Model to Text) standard. Therefore, a plugin, named *es.uclm.uml2cloud.m2t.simcan2cloud*, has been deployed to implement these transformations. An Acceleo template has then been implemented for each one of the subtransformations shown in Fig. 3.

Listing 2 shows the *generateSLAList* and *generateVMCostList* subtransformation templates implemented with Acceleo. The main purpose of the code of these templates is to obtain the number of VMs for each SLA, and the different costs for each, i.e., base, inc-priority, discount, and compensation. For this purpose, the code of the *generateSLAList* template invokes the *size* method of the property *vmcost* of each SLA in order to obtain the number of VMs. Then, the *generateVMCostList* subtransformation obtains all the prices —base, inc-priority, discount and compensation— for each one of these VMs.

```

1 [template public generateSLAList(aSLA : SLA)]
2 [aSLA.base_Component.name/] [aSLA.vmcost->size ()/]
   ↳ [aSLA.vmcost.generateVMCostList ()/]
3 [/template]
4 [template public generateVMCostList(aVMCost : VMCost)]
5 [aVMCost.vm.base_Component.name/] [aVMCost.base.getCostPerHour ()/]
   ↳ [aVMCost.inc-priority/]
   ↳ [aVMCost.discount/] [aVMCost.compensation/]
6 [/template]

```

Listing 2 *generateSLAList* and *generateVMCostList* Acceleo templates.

In Fig. 4, we show the correspondence between some components of the UML cloud model (see central components in the figure) with the code associated in the *.ini* target file (code above) and with the code in the *.ned* target file (code below). For the sake of clarity, some parts of the model and their corresponding generated code in both files have been highlighted with the same color. Furthermore, at the bottom of the figure, a legend has been included to facilitate its interpretation.

Note that the element references are only highlighted in the model because they do not generate any code. However, the elements they reference can produce their own code once translated.

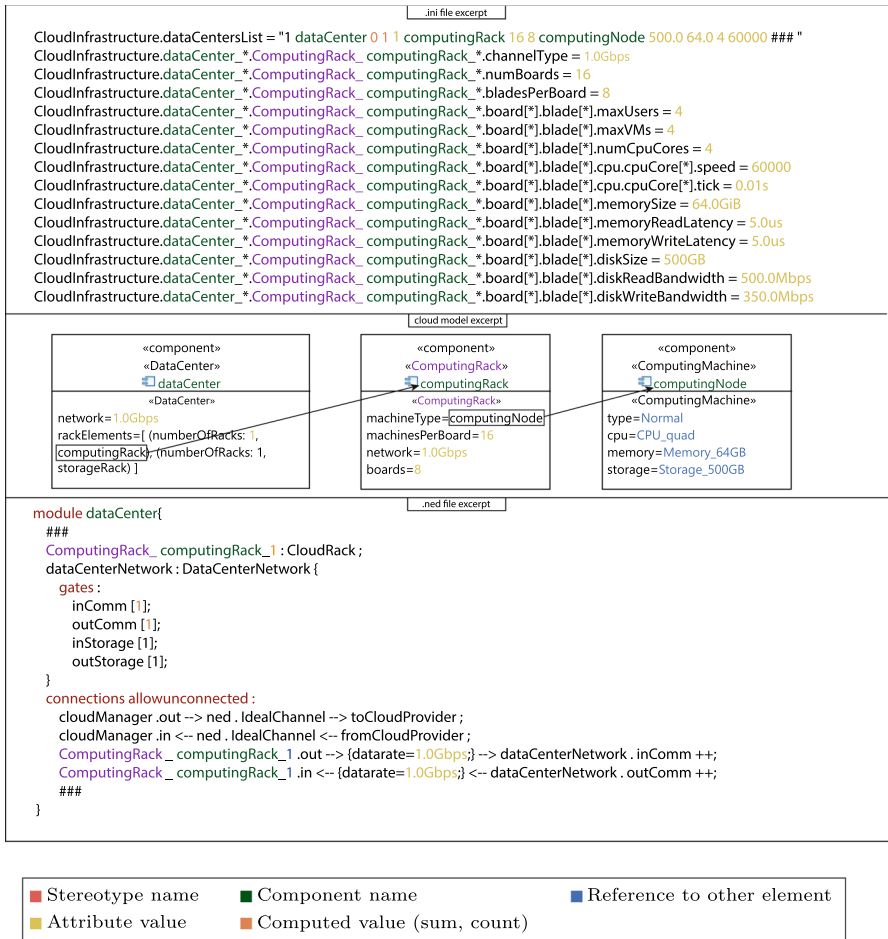


Fig. 4 Transformation example using the proposed CloudCost profile and editor

5 Evaluation

This section presents the empirical evaluation performed. In this study, we model the behavior of a wide variety of users interacting with the cloud with the goal of analyzing how the cloud architecture and the cost parameters affect the total income received.

For the sake of clarity, this study has been divided into four different sections. Firstly, Sect. 5.1 formulates different research questions, which will be answered with the results obtained. Secondly, Sect. 5.2 describes the settings used to conduct the experiments and includes a detailed description of the most relevant parameters used to model the cloud. Next, the results of each experiment are shown in Sect. 5.3. Finally, we provide a discussion of the results obtained, and the previously

formulated research questions are carefully answered using the results obtained from this study.

5.1 Research questions

The study presented in this section seeks to answer the following research questions:

RQ1: *“Is it suitable to use models of cloud infrastructures to analyze the cloud income?”*

Although there exist numerous simulators to represent the underlying behavior of the cloud, it is still a challenge to model and analyze both cost-related parameters and income.

The major difficulty of this task lies both in the complexity of the cloud and in its size. First, there are a high number of inter-related elements that must be carefully modeled to accurately represent the behavior of the cloud, such as the communication network, VMs, physical resources and resource allocation policies, to name just a few. Second, the inherently large size of the cloud requires considerable computational resources to execute the experiments in simulated environments. In order to mitigate these issues, we use a UML profile in which the cloud can be modeled using a flexible model to represent both the user behavior and the underlying cloud architecture. With these UML interaction diagrams, we capture the interactions that take place between the users and the cloud provided, identifying the messages they exchange and their contents.

In this study, we are interested in investigating those features of the cloud that significantly affect its overall income, such as the size of the cloud, the number of users accessing the cloud and the cost-related configuration of the cloud provider.

RQ2: *“How effective is the proposed approach at analyzing the income of a cloud system?”*

The profile proposed in this work represents how the users interact with the cloud and the messages they exchange. In this study, we take advantage of this UML profile to easily modify the configuration parameters of the cloud infrastructure, the user behavior, and/or the number of users trying to access the cloud. With this variety of scenarios, we investigate how these parameters affect the overall income, and more importantly, how we can tune these parameters to increase the overall income.

5.2 Experimental settings

This section describes the experimental settings used to configure the experiments, which are run on the Simcan2Cloud simulator [14, 15]. Firstly, we describe the main features of the cloud system infrastructure:

- Size: 256 / 1024 physical machines.
- Network: Ethernet Gigabit.
- Resource allocation policy: First-Fit.

Table 1 Configuration of different virtual machines (VMs)

Type	CPU cores	CU	Memory	Storage
VM_{nano}	1 core	1.0	100 GB	500 MB
$VM_{nanoRAM}$	1 core	1.0	100 GB	2 GB
VM_{nanoHD}	1 core	1.0	500 GB	500 MB
VM_{micro}	1 core	1.0	100GB	1 GB
VM_{small}	1 core	1.0	250 GB	2 GB
VM_{medium}	2 cores	2.0	500 GB	4 GB
VM_{large}	4 cores	4.0	1000 GB	8 GB

Table 2 Workload to be processed by the clouds

User instances	VMs requested by each user
1000	$5 \times VM_{small}$ for 2 h.
1000	$5 \times VM_{medium}$ for 2 h., $5 \times VM_{large}$ for 3 h.
1000	$2 \times VM_{medium}$ for 2 h.
1000	$5 \times VM_{medium}$ for 2 h.
1000	$1 \times VM_{large}$ for 10 h.
1000	$2 \times VM_{micro}$ for 2 h., $2 \times VM_{small}$ for 3 h.
1000	$2 \times VM_{micro}$ for 2 h., $2 \times VM_{small}$ for 2 h. $2 \times VM_{medium}$ for 1 h.
1000	$1 \times VM_{nanoRAM}$ for a day
1000	$1 \times VM_{nano}$ for a day
1000	$5 \times VM_{small}$ for 2 h., $5 \times VM_{medium}$ for 3 h. $5 \times VM_{large}$ for 3 hours.

- Configuration of each physical machine:

- Computing: 4-core CPU@60k MIPS.
- Memory: 64 GB of RAM.
- Storage: 2 TB of disk space.

Table 1 shows the configuration of each VM used in this study, which has been inspired by the VMs provided by Amazon EC2 [5]. Each configuration consists of the virtualized resources in a physical machine, namely computing, memory, and storage.

The workload to be processed by the clouds is shown in Table 2 and consists of a total of 10000 users, divided into 10 groups of 1000 users. Each row represents a different user role, which is defined by the resources requested, that is, the number and type of VMs rented for the specified amount of time.

We use different parameters to configure—from a cost point of view—the behavior of the cloud. The main parameters involved in the interactions are *max-SubTime*, amount of time (in hours) a user is willing to wait to obtain access to the

Table 3 Configuration of the SLAs

Type	Cost (per hour)	Inc-priority (% of the cost)	Discount (% of the cost)	Compensation (% of the cost)
VM_{nano}	0.01	50	20	10
$VM_{nanoRAM}$	0.01	50	20	10
VM_{nanoHD}	0.01	50	20	10
VM_{micro}	0.02	50	20	10
VM_{small}	0.03	50	20	10
VM_{medium}	0.05	50	20	10
VM_{large}	0.12	50	20	10

required resources; *offer*, the offer made by the cloud provider to those users that require additional time for the rented VMs (in these experiments, we assume that 90% of users accept the offer proposed by the cloud provider); *inc-priority*, the increment of cost—over the base price—for *high-priority* users; *discount*, the discount proposed by the cloud provider to the subscribed users; *compensation*, the percentage of the initial price paid by *high-priority* users that is returned due to the unavailability of physical machines for the deployment of the requested VMs; *high-priority*, the percentage of *high-priority* users (all other users are assumed to be *regular*) and *reserved*, the percentage of reserved physical machines (all other machines are *normal*, and are always active and available to both types of users).

Table 3 presents the specific SLAs that have been used for the experiments. In this table, the first column indicates the VM names, the second column the base price of each VM, the third column the *inc-priority* (as %) over the base price for *high-priority* users, and the next two columns present the *discount* for the subscribed users and the *compensation* for *high-priority* users when their requests cannot be met. The costs defined in the SLAs are based on the prices offered by Amazon AWS EC2 for T3 on-demand instances¹ at the time of carrying out the experiments. Prices tend to vary and they were therefore taken as a reference. The SLA offer is based on the public cloud model, like the one used in Amazon AWS. The cloud provider offers resources at a price and under specific conditions defined in the public SLA that all users must accept if they want to use the services. In our framework, the UML profile and the Simcan2Cloud simulator allow the definition of independent parameterized SLAs for each type of user.

5.3 Empirical study

In this section, we show the results obtained from simulating different cloud environments, with the goal of finding the best configuration for the cloud to maximize the overall income. The specific configurations that have been analyzed were chosen

¹ Amazon AWS EC2 pricing details available on the following url: <https://aws.amazon.com/es/ec2/pricing/on-demand/>.

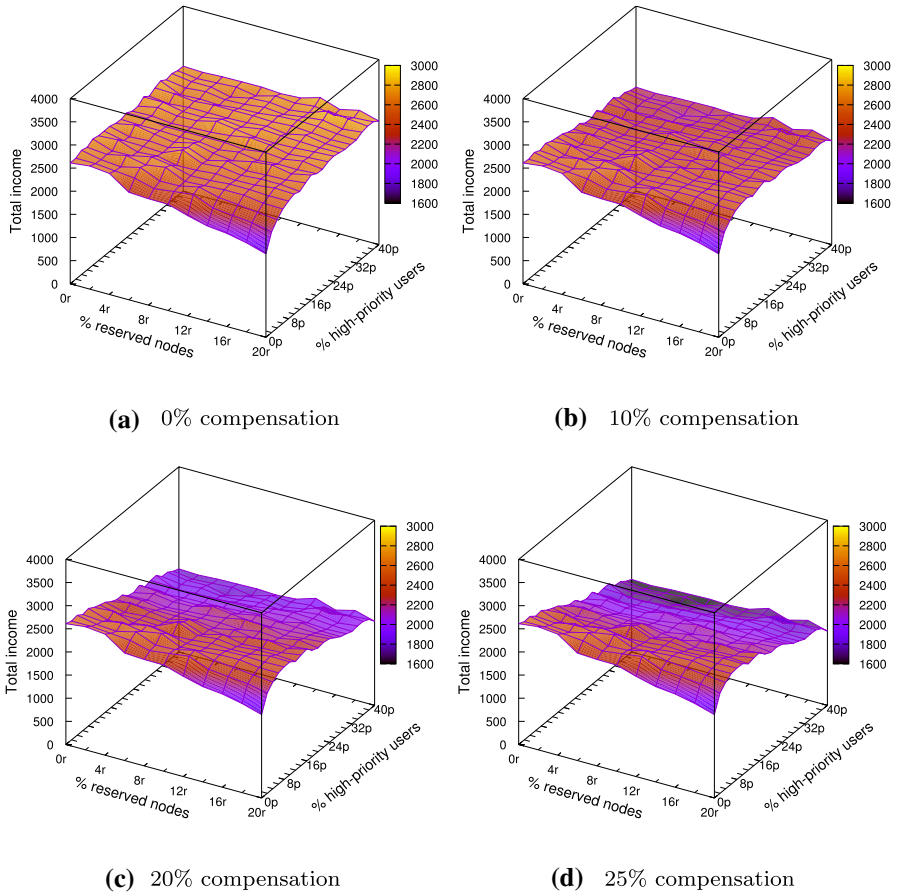


Fig. 5 Experiments varying the *compensation* in a cloud consisting of 256 machines

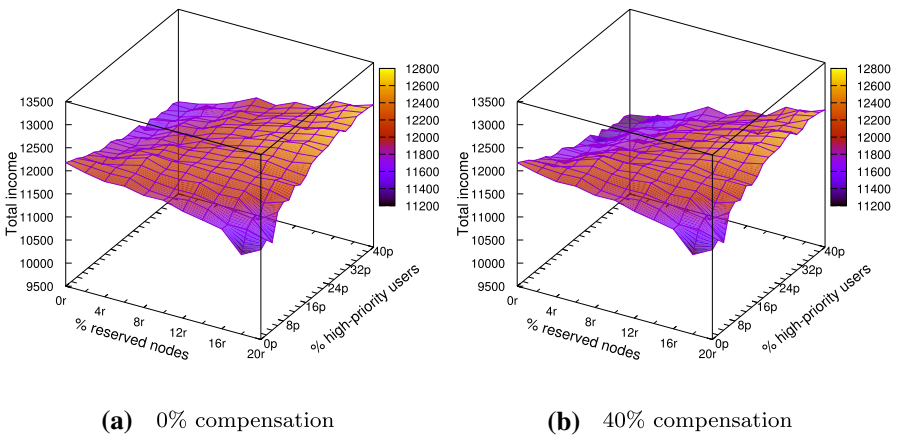


Fig. 6 Experiments varying the *compensation* in a cloud consisting of 1024 machines

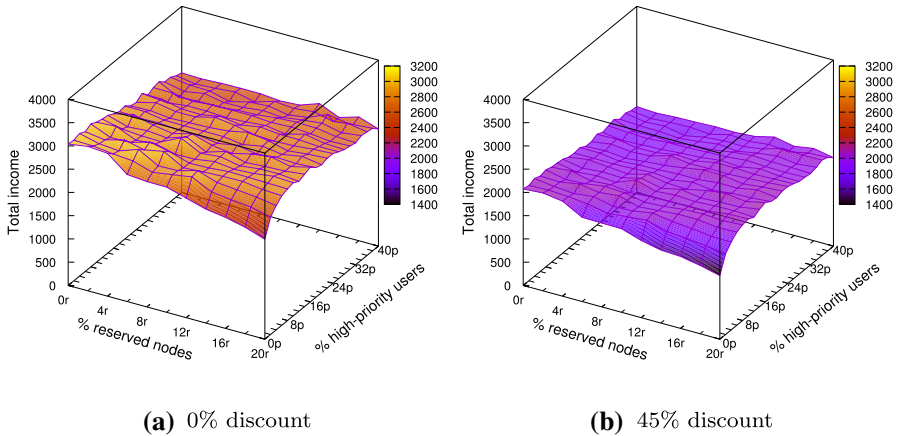


Fig. 7 Experiments varying the *discount* in a cloud consisting of 256 machines

by an expert with deep knowledge in cloud computing systems. In particular, the *maxSubTime* parameter was set to 10 hours and the *offer* parameter was initially set to 50% over the base cost; the other parameters were established according to the SLAs presented in Table 3. In this study, the idea is to use different values for a specific parameter—while the rest remain unmodified—and observe how the current configuration affects the overall income.

Initially, we analyze the impact of the *compensation* parameter on the income. The results of these experiments are depicted in Figs. 5 and 6, in which the x-axis represents the percentage of *reserved* nodes, the z-axis shows the percentage of *high-priority* users, and the y-axis represents the total income after processing the workload. We assume that the income from the cloud is zero before processing the workload. Figure 5 shows the results when the workload is processed by a cloud containing 256 physical machines. In this case, we observe that the best situation, that is, the scenario that maximizes the income, is the cloud that uses a value of 0% for *compensation* (see Fig. 5a). This means that the cloud is saturated and not capable of providing the resources for *high-priority* users, who leave the system and then receive a *compensation*, thus reducing the income. These charts show that the income decreases in proportion to the *compensation*, reaching the worst-case scenario when the *compensation* is set to 25% of the base cost (see Fig. 5d).

However, when the cloud contains 1024 physical machines (Fig. 6) we have a scenario in which the cloud is not saturated and so it is able to provide users with the requested VMs. Consequently, the *compensation* parameter has little impact on the overall income (*compensations* of 10% and 40% provide similar results). In principle, compensations should rarely occur, so little impact on income would be expected, and we would expect a greater impact from other parameters, such as *inc-priority*. Nevertheless, it is important to mention that the *inc-priority* parameter does not positively affect the income in all cases, because other parameters must be taken into account, such as the percentage of *reserved* nodes and the number of

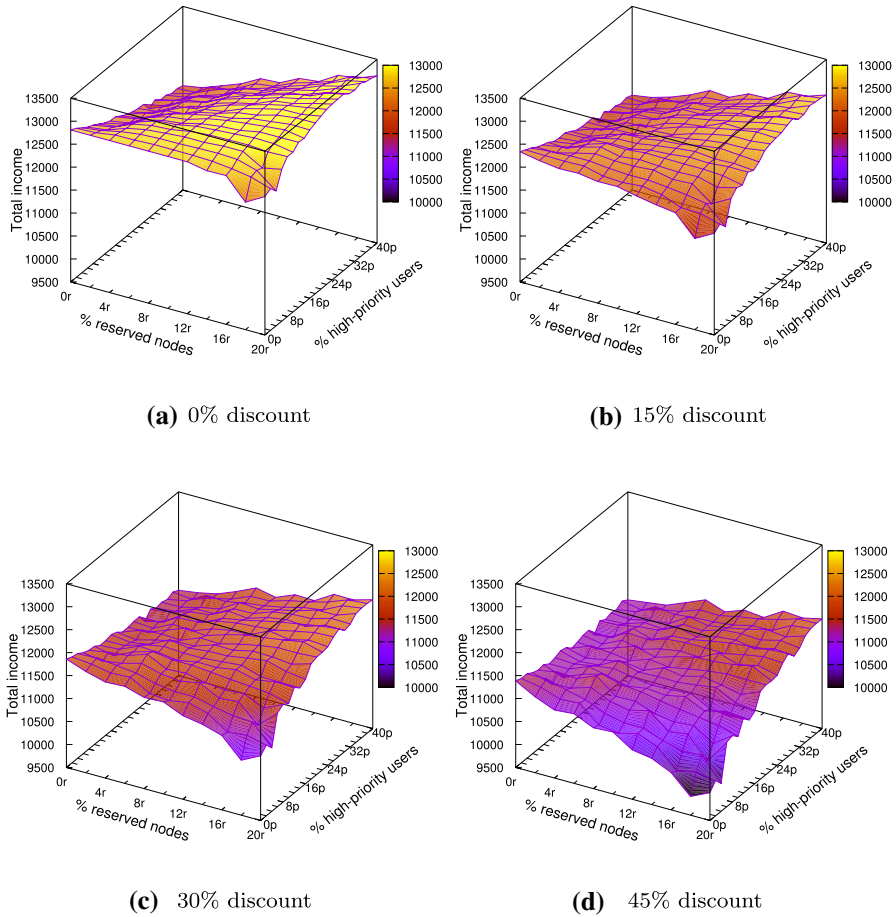


Fig. 8 Experiments varying the *discount* in a cloud consisting of 1024 machines

high-priority users. Thus, the number of *reserved* nodes must be balanced with the number of *high-priority* users. Otherwise, the overall income would be smaller.

The next experiment analyzes how the *discount* parameter affects the total income. In general, the main objective of pay-as-you-go models, which are present in most of the cloud platforms, consists of maximizing the number of users served when accessing the system. Thus, the *discount* parameters aim at encouraging users to subscribe for access to VMs if these are not available at the moment of the initial request. Figure 7 shows how the *discount* affects the total income when the cloud contains 256 physical machines. The first chart (see Fig. 7a) shows the scenario in which no discount is applied for subscribed users. In this case, the percentages of *reserved* nodes and *high-priority* users have little impact on the overall income, which is mainly due to the saturation of the cloud, therefore forcing most of the users to wait for the resources. However, when the *discount* increases, the overall

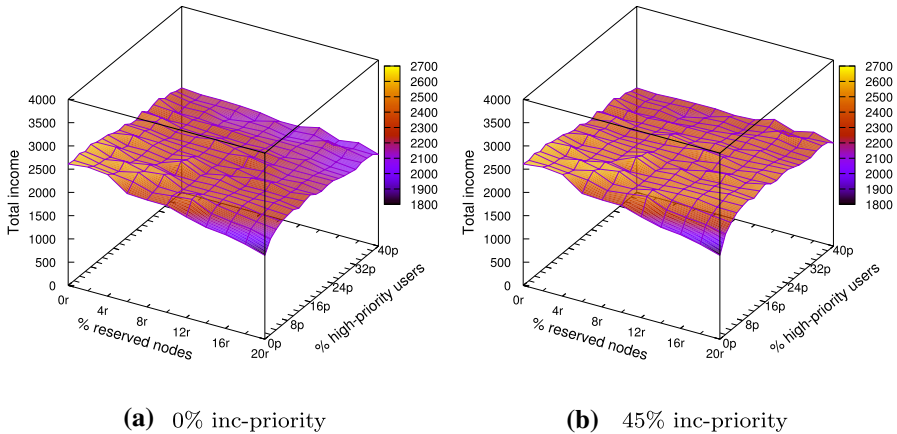


Fig. 9 Experiments varying the *inc-priority* cost in a cloud consisting of 256 machines

income is proportionally reduced. Interestingly, the shape of both charts is very similar, which represents the same tendency in this cloud.

However, when the cloud provides 1024 physical machines we observe a different scenario (see Fig. 8). First, all the charts show a different shape than the charts in Fig. 7. In this case, the cloud can process the workload, and thus, there is no saturation. The best case for the cloud provider would be that shown in Fig. 8a, where no discount is applied. In other cases, different values for *discount* are applied to the subscribed users. We can see from the charts that the income decreases in proportion to the *discount*. It is worth pointing out that the best case scenario occurs when both the percentage of *reserved* nodes and *high-priority* users are proportionally balanced. Otherwise, the income is significantly reduced.

Let us now analyze the impact of the *inc-priority* parameter, i.e., the cost for *high-priority* users. As mentioned above, *high-priority* users pay an extra fee for ensuring the availability of the VMs. For this purpose, a number of physical machines are reserved for the sole use of *high-priority* users. Then, it becomes crucial to find a balance between the number of reserved resources and the extra cost for them. If these parameters are not well balanced, the income might be compromised. The charts in Fig. 9 show that in the case of 256 machines there is no increase in the income when the percentage of *reserved* nodes or the number of *high-priority* users increases. On the contrary, we observe a drop in income (see purple areas in the corners of the charts), which means these parameters are not properly balanced, for example, by having a large number of *reserved* machines and a low number of *high-priority* users. However, the results when this experiment is carried out in a cloud containing 1024 physical machines portray a different scenario (see Fig. 10). First, the overall income is, in general, greater, but the drop in income when varying the parameters is more noticeable, especially in the corners of the charts, where extreme values are used. Second, since the cloud is not saturated, balancing the percentage of *reserved* nodes with the total number of *high-priority* users leads to a rise in the income (see Fig. 10b).

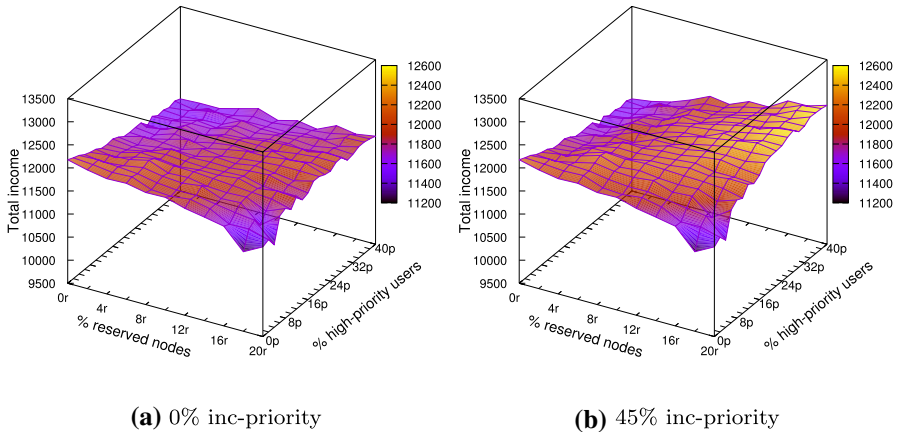


Fig. 10 Experiments varying the *inc-priority* cost in a cloud consisting of 1024 machines

Finally, we investigate how the *offer* parameter affects the cloud income. This parameter is used to allow users to extend the time requested to execute VMs before the rented time-frame expires. In this experiment, we assume that 20% of the users—from the workload—have extended the requested renting time for running the VMs. Similarly to the previous experiments, we observe that a cloud providing 256 nodes collapses and, consequently, this parameter slightly affects the cloud income (see charts in Fig. 11). However, the figure for the cloud with 1024 nodes shows that it is feasible to raise the income, not only by increasing the value of the *offer* but by using a proper value for the other parameters. Note that even in the case when the *offer* does not raise the initial cost of the VMs (set to 0%), there is a noticeable increment in the total income (see Fig. 12a) because these users can complete their executions, and so they pay for them.

5.4 Discussion of the results

In this section, we provide a detailed discussion of the results obtained and the answer to the research questions formulated in Sect. 5.1.

RQ1: “Is it suitable to use models of cloud infrastructures to analyze the cloud income?”

To answer this question, we have designed and implemented a framework, consisting of the cloud profile, a cloud editor, and a cloud simulator. With this framework, we can create and edit cloud models with specific configurations with the goal of analyzing different aspects related to the user costs and the profits obtained by the cloud provider, considering two types of user, namely *regular* and *high-priority* users. Additionally, we have developed a plug-in that implements a transformation module for translating the model of the cloud under study into the input files required by the Simcan2Cloud simulator.

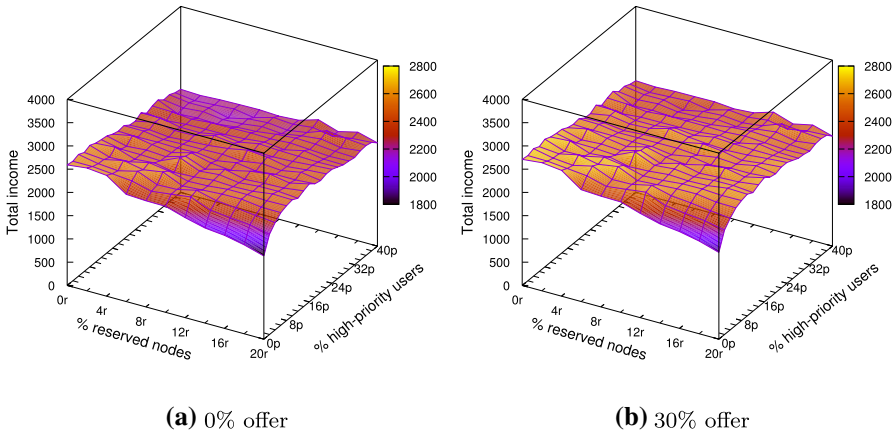


Fig. 11 Experiments varying the *offer* in a cloud consisting of 256 machines

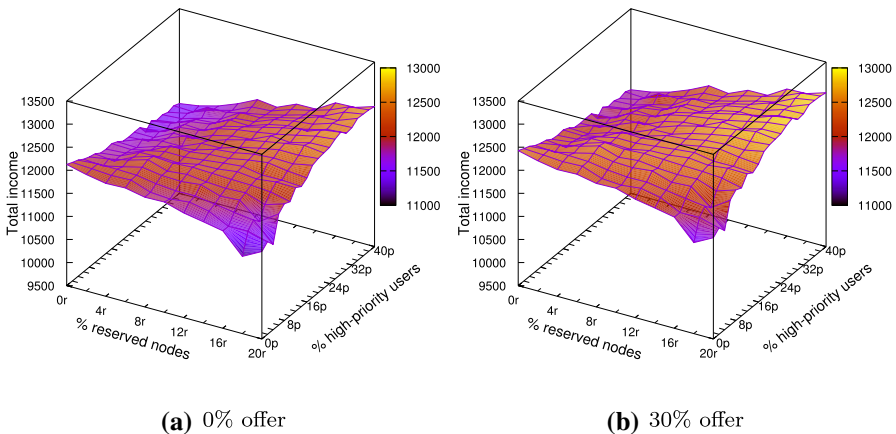


Fig. 12 Experiments varying the *offer* in a cloud consisting of 1024 machines

Thus, we can conclude that the answer to this question is *yes, it is suitable to use cloud models to analyze the income.*

RQ2: “*How effective is the proposed approach at analyzing the income of a cloud system?*”

In order to answer this question, we have performed a thorough experimental study in which two different clouds have been modeled and used to process a workload consisting of 10000 users. The results of this study, shown in Sect. 5.3, allow us to observe the considerable impact of some of the parameters we have identified on the overall income.

Firstly, we can see that the cloud containing 256 physical machines collapses and, therefore, is not capable of successfully processing the entire workload. On the contrary, the cloud containing 1024 physical machines does not become saturated and allows us to obtain greater incomes. However, we have seen that using extreme

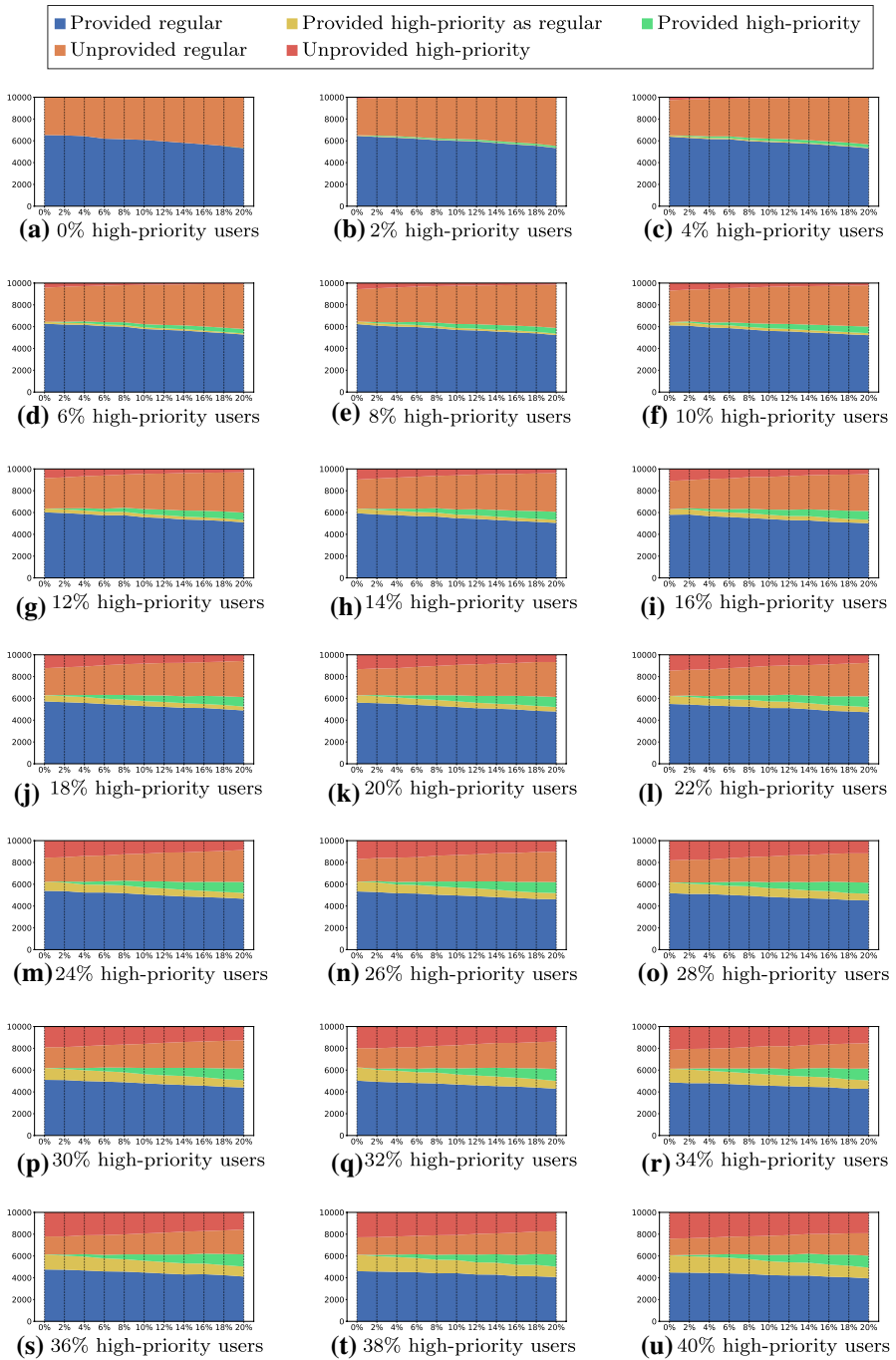


Fig. 13 Provided and unprovided users on a cloud consisting of 256 nodes

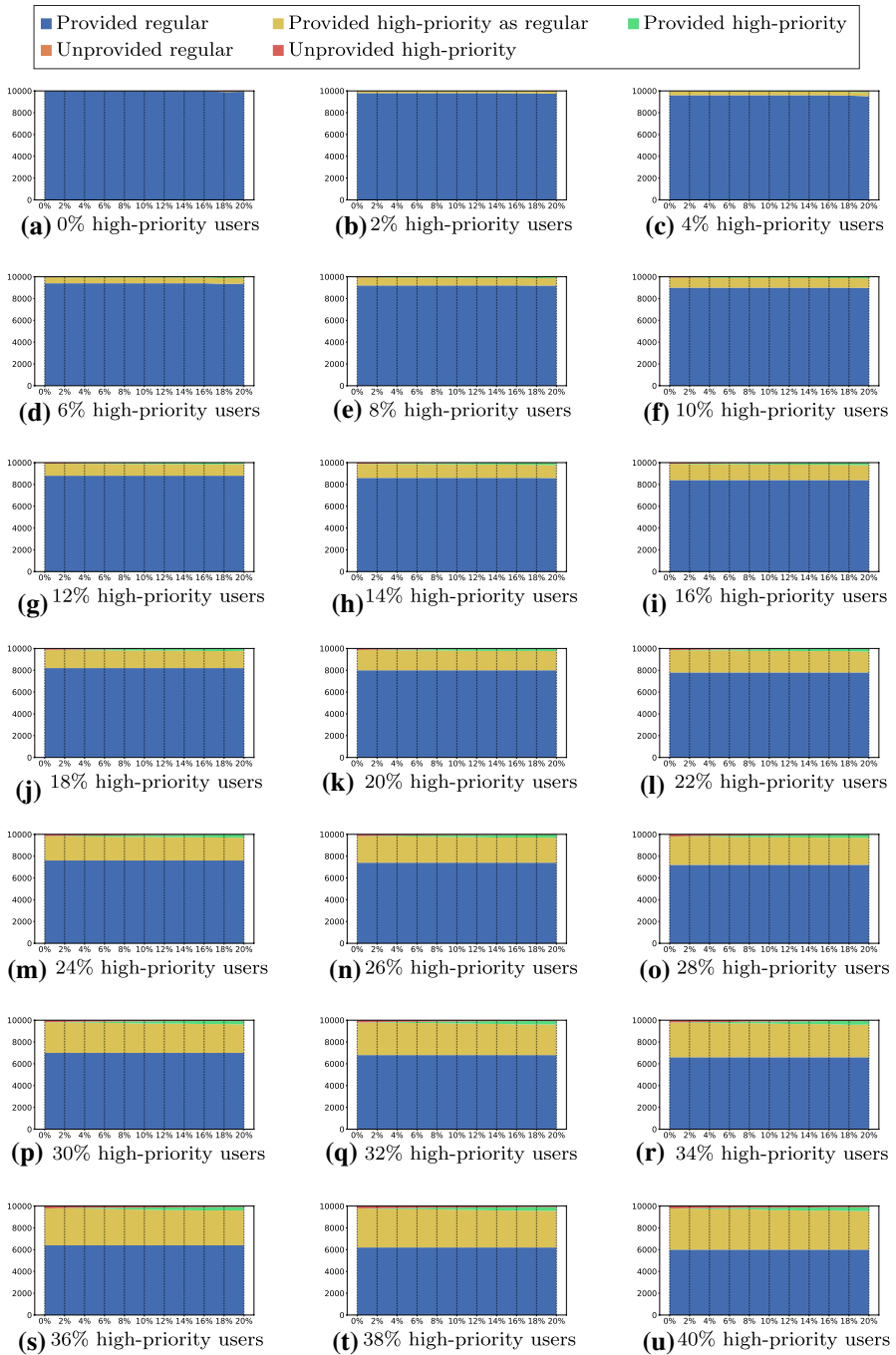


Fig. 14 Provided and unprovided users on a cloud consisting of 1024 nodes

values to configure the cloud works only in those scenarios where both the percentage of reserved machines is well balanced with the number of *high-priority* users and the cost of the VMs.

Additionally, we have generated different charts (see Figs. 13 and 14) that show the information about the users attended to for both 256 and 1024 machines. In these charts, the blue and green areas represent, respectively, the *regular* and *high-priority* users that have been successfully served. In the case of the *high-priority* users that have been served as *regular* users (using regular machines), these are shown in yellow. Finally, the *regular* and *high-priority* users that leave the system because the cloud was not able to provide the requested resources are shown in orange and red, respectively. The x-axis of these charts represents the percentage of nodes reserved for *high-priority* users, and the y-axis the number of users successfully served.

In the case of 256 machines (Fig. 13), we can see that in the best case scenario (0% of *high-priority* users), the number of *regular* users that obtain access to the requested resources is—approximately—6000 out of 10000 (shown in blue). However, this number decreases in proportion to the percentage of *high-priority* users accessing the system, reaching—approximately—4000 when 40% of the users contained in the workload are *high-priority* users. Similarly, the number of users successfully served with the requested resources follows a similar tendency, and also increases when the percentage of reserved machines is raised. It is important to note that this cloud cannot serve all the users—*regular* and *high-priority*—from the workload, even when the percentage of reserved nodes is increased. This fact can be observed from the yellow and red areas, which mean, respectively, that a significant percentage of *high-priority* users are managed as *regular* users and *regular* users cannot be attended to. On the other hand, a clear saturation of the cloud is noticed, because there is a significant number of *regular* users (see orange area) that are not able to access the requested resources. Besides, the same tendency occurs when the number of *high-priority* users increases (see red areas). In summary, this cloud of 256 machines cannot properly process the workload we have considered, so more physical machines are needed.

In contrast, Fig. 14 shows the results for a cloud with 1024 machines. In this case, almost the entire workload is successfully processed by the cloud. When there are no *high-priority* users (Fig. 14a) all users are provided with the requested VMs, which is shown in blue. When the percentage of *high-priority* users increases, the charts show a yellow area that grows in proportion to this percentage. When the percentage of *high-priority* users is significant, a slight portion of the workload is not successfully processed, which corresponds to the red and orange areas at the top of the charts.

6 Threats to validity

This section presents threats to the validity of our empirical study.

6.1 Internal threats

Internal validity refers to whether our findings, which have been formulated using the results obtained from the empirical study, represent—or not—a cause-and-effect relationship. In essence, the internal validity of our study focuses on the implementation of our experiments.

The diagrams representing the user interactions with the cloud have been designed by two experts with a deep knowledge of cloud systems. These behaviors have been coded into the Simcan2Cloud [14] simulator by three experienced programmers, including the main developer of Simcan2Cloud. We are aware that it is possible for new classes to contain errors. In order to mitigate this issue, we have conducted code inspection and run different tests to ensure the correctness of the implementation. The source code has also been reviewed by other programmers, different from the ones that carried out the implementation.

Other issues might arise due to the underlying simulator used, which might contain certain errors that could affect our findings. However, we have used Simcan2Cloud for more than a year to conduct a large number of experiments, simulating a wide spectrum of cloud scenarios, and no problem has been found in these experiments.

6.2 External threats

External validity concerns the extent to which the results of a study can be generalized.

We have used a cloud architecture using a workload consisting of ten different types of users and ten VM configurations, which have been inspired by the ones provided by Amazon EC2. Although we believe that the workload and the VM configurations are representative, there is no guarantee that the results obtained would be the same for other scenarios.

6.3 Construct threats

We evaluate the results of our experiments using well-known metrics, such as monetary cost or the number of users that fulfill a given property (i.e., users that do not access the requested resources). The simulator or the diagrams representing the behavior of users interacting with the cloud might contain defects. However, we control this threat by combining different parameters to carry out the experimental evaluation. Thus, the results obtained must be consistent with the parameters used to configure each experiment.

7 Conclusions and future work

We have proposed a framework, consisting of the cloud profile, a cloud editor, and a cloud simulator, for analyzing the profits of a cloud provider by customizing the underlying physical infrastructure and providing two different SLAs for the users (*regular* and *high-priority* users). *Regular* users can wait for the resources they need and subscribe to the cloud provider to be informed when the resources are available. In contrast, *high-priority* users need the resources immediately, and some physical resources are then reserved in order to be able to attend to their requests. We have used the cloud profile to model complex cloud scenarios, in which we can represent the underlying cloud infrastructure, the cost of the resources, and the workload submitted by the users, taking into account the SLA they have signed. A case study considering several cloud scenarios has been evaluated, with the goal of drawing conclusions about the impact of the modifications to the parameters considered. We have then concluded that the use of SLAs for reserving some physical machines is not a solution for small clouds, but the cloud provider can increase his profits by using this feature in the case of large clouds.

For future work, we have several lines of research. We will enrich the profile by including other possible SLAs, studying, for instance, the procurement schemes of Amazon Web Services (Amazon AWS [5]), as well as combinations of them. We also plan to perform further simulations with a wide spectrum of cloud configurations, not only using a different number of physical machines but also a wide range of configurations for the hardware such as, among others, the CPUs and the disk space of the hosts. Also, we plan to design new workloads representing different scenarios. Thus, we expect to obtain relevant and useful conclusions from this new study. Finally, we would like to define a formal testing framework to complement our approach. In this line, as a starting point, we will take the recent work on formal testing of distributed systems [34, 35], in which probabilities can be used to guide the choice between options with similar performance/cost ratios.

Acknowledgements This work was supported by the Spanish Ministry of Science and Innovation (co-financed by European Union FEDER funds) projects “FAME (Formal modeling and advanced testing methods. Applications to medicine and computing systems),” reference RTI2018-093608-B-C32 and “MASSIVE (Engineering adaptive software by and for the people in a highly connected world)” under reference and RTI2018-095255-B-I00. There was also support from the Junta de Comunidades de Castilla-La Mancha project SBPLY/17/180501/000276/01 (cofunded with FEDER funds, EU), the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with the Complutense University as part of the Program to Stimulate Research for Young Doctors in the context of the V PRICIT (Regional Programme of Research and Technological Innovation) under grant PR65/19-22452, the Region of Madrid under Grant FORTE-CM, S2018/TCS-4314 and the University of Castilla-La Mancha (cofunded with FSE funds, EU) under the announcement 2018/12504 published in the DOCM.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is

not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. SRG (2020) About us. <https://www.srgresearch.com/about>
2. SRG (2020) Half-Yearly Review Shows \$150 Billion Spent on Cloud Services and Infrastructure. <https://xurl.es/7wlin>
3. Kulkarni G, Sutar R, Gambhir J (2011) Cloud computing-infrastructure as service amazon EC2. *Int J Eng Res Appl* 2(1):117–125
4. Chavan P, Kulkarni G (2013) PaaS Cloud. *Int J Comput Sci Inf Secur* 1:21–26
5. Amazon (2020) Amazon Elastic Compute Cloud. <http://aws.amazon.com/ec2/>
6. Microsoft Azure (2017) <http://azure.microsoft.com>
7. Krishnan SPT, Ugia Gonzalez JL (2015) Building your next big thing with google cloud platform: a guide for developers and enterprise architects, 1st edn. Apress, USA
8. Snowflake (2020) Your cloud data platform. <https://www.snowflake.com/>
9. Veeam (2020) Veeam Company. <https://www.veeam.com/>
10. European Union (2019) The European Cloud Initiative. <https://ec.europa.eu/digital-single-market/en/european-cloud-initiative>
11. OMG (2015) UML Specification Version 2.5. <http://www.omg.org/spec/UML/2.5>
12. OMG (2014) UML Profile Specifications. <http://www.omg.org/spec/#Profile>
13. OMG. MOF (2008) Model to text transformation language (MOFM2T). p v1.0
14. Bernal A, Cambronero ME, Núñez A, Cañizares PC, Valero V (2019) Improving cloud architectures using UML profiles and M2T transformation techniques. *J Supercomput* 75(12):8012–8058
15. Bernal A, Cambronero ME, Valero V, Núñez A, Cañizares PC (2019) A framework for modeling cloud infrastructures and user interactions. *IEEE Access* 7:43269–43285
16. Cambronero ME, Bernal A, Valero V, Cañizares PC, Núñez A (2021) Profiling SLAs for cloud system infrastructures and user interactions. *PeerJ Comput Sci* 7:e513
17. García-García JA, Enriquez JG, Ruiz M, Arévalo C, Jiménez-Ramírez A (2020) Software process simulation modeling: systematic literature review. *Comput Stand Interfaces* 70:103425
18. Kamali A, Mohammadi S, Barforoush AA (2014) UCC: UML profile to cloud computing modeling: using stereotypes and tag values. In: 7th International Symposium on Telecommunications (IST'2014), pp 689–694
19. Bergmayr A, Troya Castilla J, Neubauer P, Wimmer M, Kappel G (2014) UML-based cloud application modeling with libraries, profiles, and templates. In: *CloudMDE@ MoDELS*, pp 56–65
20. Soni A, Hasan M (2017) Pricing schemes in cloud computing: a review. *Int J Adv Comput Res* 7:60–70
21. Chen S, Lee H, Moinzadeh K (2018) Pricing schemes in cloud computing: utilization-based versus reservation-based. *Prod Oper Manag* 28:82–102
22. Cong P, Li L, Zhou J, Cao K, Wei T, Chen M, Hu S (2018) Developing user perceived value based pricing models for cloud markets. *IEEE Trans Parallel Distrib Syst* 29(12):2742–2756
23. Chi Y, Li X, Wang X, Leung VCM, Shami A (2017) A fairness-aware pricing methodology for revenue enhancement in service cloud infrastructure. *IEEE Syst J* 11(2):1006–1017
24. Wang T, Zhou J, Zhan G, Wei T, Hu S (2020) Customer perceived value- and risk-aware multiserver configuration for profit maximization. *IEEE Trans Parallel Distrib Syst* 31(5):1074–1088
25. Mei J, Li K, Tong Z, Li Q, KenLi Li (2019) Profit maximization for cloud brokers in cloud computing. *IEEE Trans Parallel Distrib Syst* 30(1):190–203
26. Fabra J, Ezpeleta J, Álvarez P (2019) Reducing the price of resource provisioning using EC2 spot instances with prediction models. *Futur Gener Comput Syst* 96:348–367
27. Manvi SS, Shyam GK (2014) Resource management for infrastructure as a service (IaaS) in cloud computing. *J Netw Comput Appl* 41:424–440
28. Li C, Tang J, Ma T, Yang X, Luo Y (2020) load balance based workflow job scheduling algorithm in distributed cloud. *J Netw Comput Appl*, 152, In Press

29. Khan AA, Zakarya M, Khan R, Rahman IU, Khan M, Khan AUR (2020) An energy, performance efficient resource consolidation scheme for heterogeneous cloud datacenters. *J Netw Comput Appl*, 150, In Press
30. Herzfeldt A, Floercke S, Ertl C, Krcmar H (2018) The role of value facilitation regarding cloud service provider profitability in the cloud ecosystem, pp 121–142
31. Núñez A, Fernández J, Filgueira R, García F, Carretero J (2012) SIMCAN: a flexible, scalable and expandable simulation platform for modelling and simulating distributed architectures and applications. *Simul Model Pract Theory* 20(1):12–32
32. Núñez A, Fernández A, García JD, García F, Carretero J (2010) New techniques for simulating high performance MPI applications on large storage networks. *J Supercomput* 51(1):40–57
33. Eclipse Foundation (2020) Inc. Acceleo. <https://www.eclipse.org/acceleo/>
34. Hierons RM, Merayo MG, Núñez M (2018) Bounded reordering in the distributed test architecture. *IEEE Trans Reliab* 67(2):522–537
35. Hierons RM, Núñez M (2017) Implementation relations and probabilistic schedulers in the distributed test architecture. *J Syst Softw* 132:319–335

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.