

RESEARCH

Open Access



# Decision making in cloud environments: an approach based on multiple-criteria decision analysis and stochastic models

Julian Araujo<sup>1\*</sup>, Paulo Maciel<sup>1</sup>, Ermeson Andrade<sup>2</sup>, Gustavo Callou<sup>2</sup>, Vandi Alves<sup>1</sup> and Paulo Cunha<sup>1</sup>

## Abstract

Cloud computing is a paradigm that provides services through the Internet. The paradigm has been influenced by previously available technologies (for example cluster, peer-to-peer, and grid computing) and has now been adopted by almost all large organizations. Companies such as Google, Amazon, Microsoft and Facebook have made significant investments in cloud computing, and now provide services with high levels of dependability. The efficient and accurate assessment of cloud-based infrastructure is fundamental in guaranteeing both business continuity and uninterrupted public services, as much as is possible. This paper presents an approach for selecting cloud computing infrastructures, in terms of dependability and cost that best suits both company and customer needs. We use stochastic models to calculate dependability-related metrics for different cloud infrastructures. We then use a Multiple-Criteria Decision-Making (MCDM) method to rank the best cloud infrastructures, taking customer service constraints such as reliability, downtime, and cost into consideration. A case study demonstrates the practicability and usefulness of the proposed approach.

**Keywords:** Cloud computing, MCDM, Dependability, Stochastic models

## Introduction

Cloud computing has enabled the emergence of several service-oriented resources, such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [27]. The development and use of such cloud-based services have resulted in an increased number of users, and higher levels of data produced by different devices and applications. Several corporations and institutions have shown an interest in cloud computing, and because of this many cloud computing platforms have been proposed. Google, Amazon, Microsoft, and Facebook are examples of companies that are investing heavily in cloud computing services [30, 37].

Cloud computing has grown rapidly, and has gained popularity because it offers several benefits including on-demand self-service, virtualization, geographic distribution, and resilience [3]. These benefits are particularly

attractive because they can offer flexibility guarantees for customer service constraints such as downtime and cost, which are negotiated by cloud providers through Quality of Service (QoS) guarantees. However, providing cloud services according to the customer needs and specific constraints remains a challenge. Parameters such as reliability, capacity-oriented availability and cost are relevant factors in the negotiation of such services [6, 10]. Therefore, an efficient and accurate assessment of cloud infrastructures considering availability, reliability and cost requirements is fundamental in allowing customers to identify a cloud infrastructure that suits their needs and preferences.

To provide uninterrupted cloud services, cloud managers must evaluate and improve dependability aspects of cloud infrastructures, such as availability and transaction loss; this is because users require a reasonable level of confidence in such infrastructures to efficiently plan and operate their business [19]. Some services may be considered mission-critical, and depending on the number of data operations involved, it may be essential to

\*Correspondence: [cjma@cin.ufpe.br](mailto:cjma@cin.ufpe.br)

<sup>1</sup>Federal University of Pernambuco (UFPE), Informatics Center (Cin), Recife, Brazil

Full list of author information is available at the end of the article

deploy redundancy strategies [24]. Such strategies can lead to avoidance of outage due to issues such as database deadlock, data loss, or network failure. Cloud outages can cause significant financial losses to an organization, and in extreme cases may result in the failure of the business [4].

In this context, dependability models like Reliability Block Diagrams (RBDs) and Stochastic Petri Nets (SPNs) can be useful when comparing cloud infrastructures [7, 28]. Cloud infrastructures differ one from another in many aspects, and this results in significant challenge for cloud users attempting to identify the infrastructure that best suits their needs [34, 44]. There are always trade-offs when considering different cloud alternatives; for example, robust cloud infrastructures may result in unnecessary costs to guarantee against events that are very unlikely to happen, while simple infrastructures may result in loss of critical data. MCDM methods, which consist of techniques to solve such multi-criteria problems, are essential because they can assist cloud users in choosing the best cloud infrastructure, and can take into account multiple criteria like capacity-oriented availability, reliability, downtime, or cost.

MCDM methods are designed to analyze and give recommendations on situations involving a large number of alternatives and conflicting criteria. In [33], the authors presented a case study to compare different MCDM methods in order to select IaaS services. Garg et al. [14] proposed a framework based on an Analytic Hierarchy Process (AHP) to rank cloud providers. In [23], the authors presented a multi-attribute group decision-making (MAGDM) approach for selection cloud providers. Differently from these works, we present an approach based on a MCDM method and stochastic models to evaluate, rank and find a set of optimal cloud environments considering dependability (e.g.: capacity-oriented availability and reliability), and cost requirements.

The process of choosing a cloud infrastructure can be slow, tedious and costly; it can also generate conflicts of interest considering a set of alternatives. In recognition of the importance of making an appropriate cloud infrastructure choice, we propose a novel approach which implements an Multiple-Criteria Decision-Making (MCDM) method to rank the best infrastructure, and takes customer service constraints such as dependability and cost into consideration. Although there are several methods for multi-criteria decision-making, we adopted the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) method [11] due to its simplicity and easiness to apply. The results allow cloud customers to identify and choose the cloud infrastructure that best suits their needs, in a fast and efficient manner. Specifically, our contributions are:

- We design and implement a strategy that combines a decision-making method, with the capacity of stochastic models to obtain dependability-related metrics (like reliability and capacity-oriented availability).
- Our modeling strategy is based on hierarchical and heterogeneous modeling for planning cloud infrastructures, which allows the evaluation of cloud infrastructures with complex redundant mechanisms and maintenance policies.
- We developed a tool (called MiPACE) to support the planning of cloud infrastructures which consider customer service constraints, and assists in the decision-making process.
- We demonstrate the feasibility of our approach by showing real case scenarios and identify a set of ideal cloud infrastructures.

The remaining sections are organized as follows. “**Background**” section describes some general concepts. “**Related work**” section presents the related work. “**Adopted strategy and base cloud architecture**” section shows the proposed approach for ranking cloud infrastructures according to customers needs, and details the base cloud architecture adopted by this paper. “**Hierarchical models and cost equations**” section illustrates the hierarchical modeling process and cost equations. “**MiPACE: a multi-criteria tool for planning and analysis of cloud environments**” section presents the developed tool used to support the decision-making process. “**Results and discussion**” section illustrates the proposed approach through a real-world case study. “**Final remarks**” section concludes the paper and presents future directions.

## Background

This section introduces fundamental concepts on multiple-criteria decision-making, dependability modeling, stochastic Petri net, and reliability block diagram.

### Multiple-criteria decision-making

In our daily lives, we do not consider just one criterion when making a decision, but rather compare and evaluate more than one alternative simultaneously. When purchasing a cloud service for example, security, processing power, networking throughput, and storage capacity may all be considered as main criteria. It would be unusual for the cheapest cloud service to have the highest reliability and unlimited storage, and it is necessary to evaluate all potential impact when making decisions that involve long-term commitment and budget allocation. Thus, companies must consider multiple criteria when determining the best cost-benefit ratio. Multiple-Criteria Decision-Making (MCDM) methods have been developed to support the decision-making process in

solutions that exhibit multiple conflicting criteria, and thus provide techniques for finding a set of optimal solutions.

A large number of MCDM techniques have been proposed, each with different perspectives and theories. Some techniques are used to solve ranking problems, such as the Analytic Hierarchy Process (AHP), Analytic Network Process (ANP), Elimination and Choice Expressing Reality (ELECTRE III), and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) approaches [11]. Other approaches adopted monitoring and historical data combining with ranking techniques for decision making [15]. In this paper, the concept of the TOPSIS method has been adopted for ranking cloud infrastructures. TOPSIS is a useful technique for ranking and selecting a number of externally determined alternatives, using distance measures such as Euclidean, Manhattan and Minkowski [39]. These distances measures order alternative solutions from the best to the worst by means of scores or pairwise comparisons. They are based on five stages, where the first step groups the set of alternative solutions when taking the defined criteria into account. In the second step, the values representing each criterion are normalized; this allows all criteria to be treated in a similar way, independent of the metric adopted. Next, the criteria can be weighted and the distances between each solution are calculated, taking an anti-ideal and an ideal point (optimal solution) into consideration. In the fourth step, the relative closeness to the ideal solution is calculated. In the last step, a set of alternatives is ranked according to the relative closeness [18].

### Dependability modeling and evaluation

The dependability [21] of a system is defined as its justifiably trusted ability to deliver a set of services. Dependability requirements encompass the concepts of reliability, availability, maintainability, performability, and testability. This paper focuses on availability and reliability modeling, and analysis of cloud infrastructures. Availability is the probability that the system is working (even if not at its full capacity) over time, whereas reliability is the probability that the system will deliver a set of services over a given period of time [21, 25]. The steady state availability (A) may be calculated by Eq. 1:

$$A = \frac{MTTF}{MTTF + MTTR} \quad (1)$$

where MTTF and MTTR denote the mean time to failure and mean time to repair, respectively.

For any given time period represented by the interval  $(0, t)$ ,  $R(t)$  is the probability that the component has continued to function (i.e. has not failed) from 0 until  $t$ .

When an exponentially distributed time to failure (TTF) is considered, reliability is represented by

$$R(t) = \exp \left[ - \int_0^t \lambda(t') dt' \right] \quad (2)$$

where  $\lambda(t')$  is the instantaneous failure rate.

We also adopt the Capacity-Oriented Availability (COA) as [25, 42]. COA takes into account how much of a service provided by a system is delivering, therefore, does not consider only states of availability or unavailability, but the impact of these conditions in service delivery. The COA calculation considers  $pc_i$  as operational processing capacity or the amount of resource available at any state  $s_i$ .  $\pi_i$  is the probability of being at state  $s_i \in S$ , where  $S$  is the set of reachable states. And the maximum capacity of the system is  $N$ . Thus, we can calculate the COA by Eq. 3.

$$COA = \frac{\sum_{s_i \in S} pc_i \times \pi_i}{N} \quad (3)$$

### Redundancy techniques

In several application domains, different techniques have been adopted to increase the dependability of systems. These techniques are traditionally classified into four groups: fault prevention, fault removal, fault forecasting, and fault tolerance [38]. Unlike other techniques, fault tolerance (redundancy) aims to provide correct service delivery even in the presence of faults. Redundancy refers to extra resources that are not necessary for the execution of the faultless task, but must be applied if faults occur to guarantee the service delivery.

The redundancy techniques for fault tolerance include active-standby and active-active redundancy [5]. In an active-active redundancy mechanism, both the main elements (e.g., resource and service) and the redundant elements are permanently active. The users do not perceive the occurrence of faults, nor does performance degradation take place. In contrast, active-standby mechanisms are characterized by fault detection followed by recovery actions, which require extra processing time. This type of strategy uses two component types: active, and standby. The active module usually provides the service for all environments; if the active module fails, the standby component assumes control. Standby modules are classified as hot, warm, or cold, depending on the level of service restoration [24].

### Stochastic Petri net

Petri nets are very well suited for modeling several system types. This is because concurrency, synchronization, communication mechanisms, and deterministic and probabilistic delays, are naturally represented. In general, Petri nets are a bipartite directed graph, in which places (represented by circles) denote local states, and transitions (depicted as rectangles) represent actions. Arcs (directed

edges) connect places to transitions, and vice versa. The original Petri Net does not have the notion of time for analyzing performance and dependability; the introduction of event durations results in a timed Petri Net.

Stochastic Petri nets (SPN) [26] is a special type of timed Petri Net, which allows the association of probabilistic delays with transition, by using exponential distribution. It is a high-level model which allows automatically to generate and evaluate Continuous Time Markov Chain (CTMC) [42]. This characteristic is particularly useful when the system's state space is large and/or system component's interactions are complex. Besides, SPN may also be evaluated through simulation. Simulation may be the alternative when non phase-type distribution [42] are required and/or the system state space is infinity.

### Reliability block diagram

A Reliability Block Diagram (RBD) [12] is a combinatorial model, initially proposed as a technique for calculating the reliability of a system by using block diagrams. The technique has also been extended to calculate other dependability metrics, such as availability [21, 24, 25]. RBD may be a model of choice for computing availability and reliability related metrics for passive redundant mechanism and/or independent component systems [25]. In RBD, model are usually obtained by serial and parallel composition of components and subsystems.

In an arrangement series, the whole system is no longer operational if a single component fails. This means that all components must be operational for the serial system to succeed. If a system with  $n$  independent components is considered, the reliability (instantaneous availability or steady-state availability) is obtained by the product of component's reliabilities (instantaneous availability or steady-state availability). In a parallel arrangement, the whole system is considered operational even if only a single component is operational, because there are a total of  $n$  possible success paths. For a system with  $n$  independent components, the unreliability (instantaneous unavailability or steady-state unavailability) is obtained by the product of component's unreliability (instantaneous unavailability or steady-state unavailability).  $k$ -out- $n$  redundancy may also be represented by RBDs.  $k$ -out- $n$  RBD models allows you to represent more general compositions than simple series or parallel configurations. Actually, simple series or parallel configurations are special cases of  $k$ -out- $n$  compositions [24, 25, 31, 42].

### Failure critical index

In general, component importance ranking indicates the impact of a particular component on the overall system reliability. Based on certain system characteristics, various measures are calculated to estimate the component importance, and this often relates the contribution of a

component to the system failure. Birnbaum introduced this concept, which can be considered one of the most widely used reliability importance indices [21]. The Birnbaum importance of a component  $i$  is equal to the degree of improvement in system reliability, when the reliability of the component is increased by one unit [21]. In other words,  $RI$  (reliability importance) is a partial derivative of system reliability with respect to the failure rate of each individual component [17].

The  $RI$  of component  $i$  can be computed as

$$I_i^B = R_s(1_i, \mathbf{p}^i) - R_s(0_i, \mathbf{p}^i) \quad (4)$$

where  $I_i^B$  is the reliability importance of component  $i$ ,  $\mathbf{p}^i$  is the component reliability vector with the  $i$ th component removed,  $0_i$  represents the condition when component  $i$  fails, and  $1_i$  describes the condition when  $i$  is working.

$I_i^B$  depends on the structure of the system and the reliability of the other components. The  $RI$  of a component  $i$  is determined by the reliability of the other components, excluding  $i$  [32].

### Related work

The increasing number of cloud platforms added to the competition among various cloud providers, has resulted in a situation whereby customers may find selection of a dependable and cost-effective cloud infrastructure difficult. In this context, several approaches have been proposed to assist cloud customers with identification of a suitable cloud infrastructure.

Rehman [35] presented a cloud selection approach based on historical QoS to rank cloud services; the proposed approach captures variations in each time-slot, and a service selection decision is then made. All decisions are then aggregated to find the best option. Lee et al. [22] proposed a hybrid multi-criteria decision-making model for a cloud service selection problem using balanced scorecard (BSC), fuzzy Delphi method (FDM) and fuzzy analytical hierarchy process (FAHP). Sachdeva et al. [36] combined a hybrid TOPSIS method with an intuitionistic fuzzy set to select appropriate cloud solutions to manage big data projects in a group decision-making environment. Garg et al. [14] proposed a framework called SMI-Cloud, which compares different cloud providers based on user requirements. The framework considers a set of attributes (e.g. accountability, agility, assurance of service, performance, cost, and usability) when prioritizing and ranking the best services, with the ranking mechanism based on an Analytic Hierarchy Process (AHP). Liu et al. [23] presented a multi-attribute group decision-making (MAGDM) approach for the process of choosing an adequate cloud service vendor. This approach considered objective attributes (i.e., cost and time), as well as subjective attributes such as TOE (Technology, Organization, and Environment). To demonstrate the usefulness of the

approach, a hypothetical example was given. Differently from these works that use data from other works or estimate data from case studies to ranking, we use the results obtained from the proposed hybrid approach to generate a set of optimal solutions. The approach combines RBD and SPN models to represent and evaluate cloud infrastructures with complex redundant mechanisms and maintenance policies.

Other work has evaluated the dependability of cloud infrastructures. Wei et al. [45] presented a hierarchical approach that combines reliability block diagrams and general stochastic Petri nets in evaluating the dependability of a virtual data center. Andrade et al. [1] developed a framework for transforming elements of SysML diagrams into deterministic and stochastic Petri nets. The work focused on modeling and analysis of cloud service management, with the aim of maximizing the use of cloud computing resources at the lowest possible cost. Sousa et al. [41] proposed a modeling strategy for planning cloud infrastructure when considering dependability and cost requirements. This approach is based on hierarchical and heterogeneous modeling that combines combinatorial and state-space models to represent and evaluate cloud infrastructures. Dantas et al. [9] described stochastic models for evaluating private cloud architectures, and presented a comparative cost study of public and private cloud providers. Despite the fact that these works are interesting, they only concerned with evaluating scenarios and presenting a comparison of the quantitative results.

The works discussed above has attempted to solve one side of the problem only: either the cloud selection perspective, or the dependability and cost evaluation aspect. However, there are various kinds of cloud environments with conflicting requirements that needs scientific approaches to judge which one should be chosen. To fill this research gap, a strategy that combines the need to rank different service constraints with the capacity to study alternative cloud infrastructures, is presented.

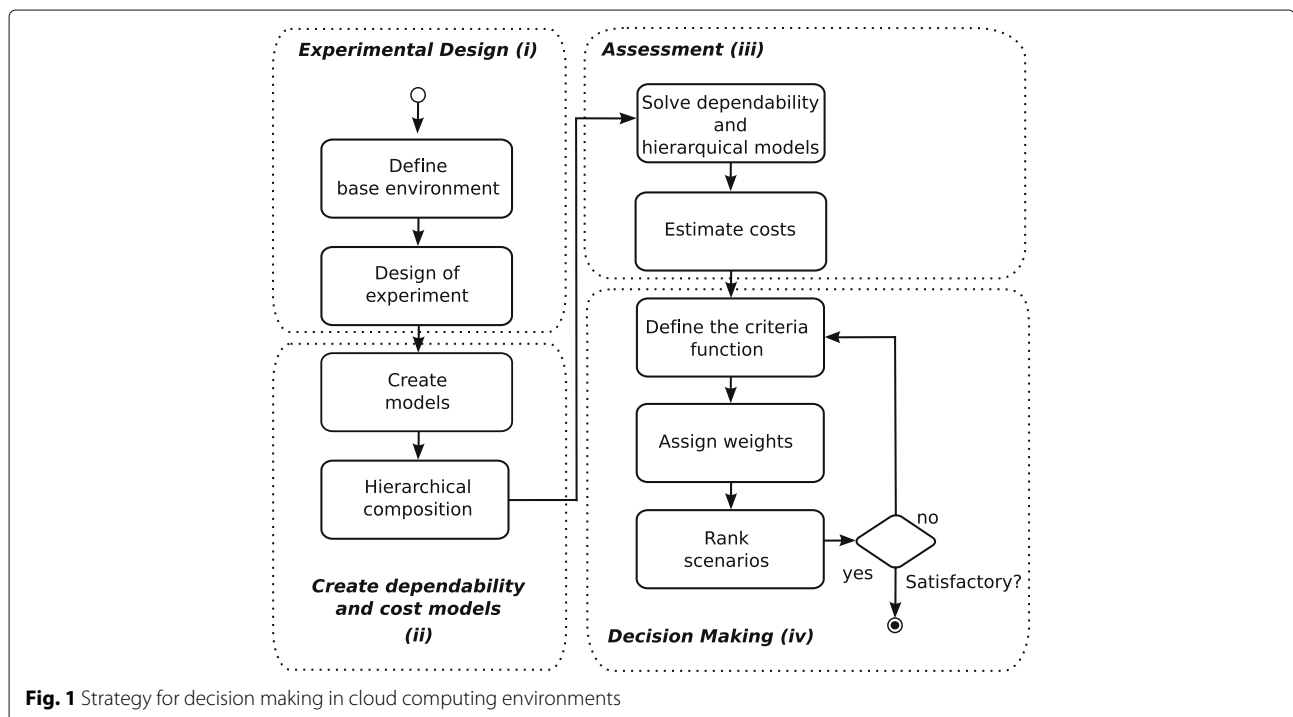
### Adopted strategy and base cloud architecture

This section first introduces the proposed strategy for modeling, analysis, and ranking of cloud computing environments. After that, the base cloud architecture adopted for this study is detailed.

#### A strategy for decision making in cloud computing environments

The strategy is based on stochastic models and an MCDM method to rank a set of cloud infrastructures, taking into account availability, capacity-oriented availability, reliability and cost requirements. The proposed strategy can be used by service providers or individuals who are interested in building and selecting their own cloud computing environments. Figure 1 illustrates the proposed strategy. The macro activities consist of (i) experimental design, (ii) creation of availability and cost models, (iii) assessment, and (iv) the decision-making process.

Experimental design (i): This activity comprises two steps: defining the base cloud environment, and designing



the experiment. The first step determines the nature of the cloud environment in terms of its components and their interactions, and defines a base cloud environment. The experiment is designed to investigate the effects of variations in one or more parameters on the base cloud environment. This is accomplished by generating distinct scenarios from the base cloud environment (e.g. redundant nodes and repairing service), and investigating the impact of these modifications on adopted metrics such as reliability and capacity-oriented availability.

Create dependability and cost models (ii): The modeling strategy comprises two steps: creation of models, and hierarchical composition. The first step aims to identify a set of individual components from the base infrastructure to be modeled through RBDs. These models are useful to analyze the reliability/availability of simple and complex systems. They can also be used to model variations in the base architecture defined from the design of experiments, such as the redundancy of nodes or virtual machines. Nevertheless, RBDs cannot easily handle detailed failure/repair behavior, and SPNs are therefore adopted to model complex redundant mechanisms and maintenance policies. We combine the strongest advantages of these models to perform the analysis and constitutes a hierarchical model. We also propose equations for estimating the costs of the cloud environment, considering associated maintenance and operational costs.

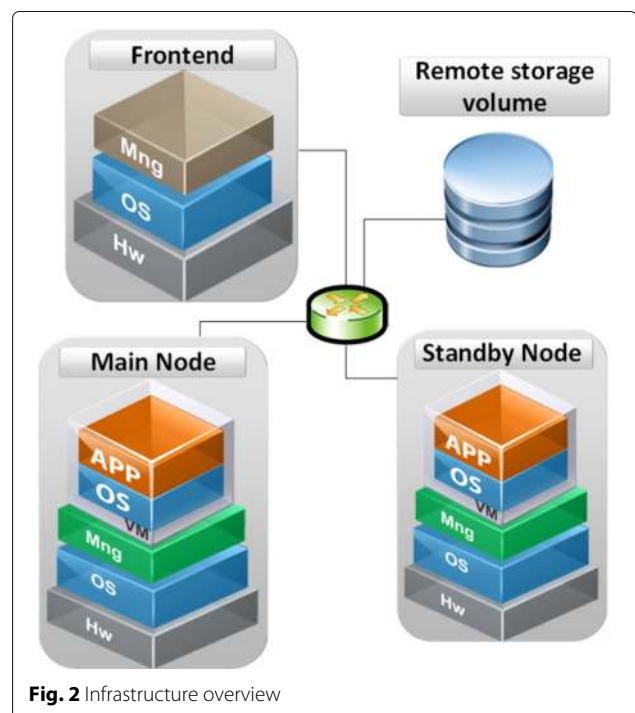
Assessment (iii): This is a macro activity in which a hierarchical model, comprising RBD and SPN models, is used to evaluate the impact that different redundant mechanisms and maintenance policies have on the steady-state availability and reliability of an environment. The hierarchical model solution is computed by passing the outputs of the SPN models (the lower-level sub-models) as inputs to the higher level sub-models represented by the RBDs. Cost equations are also solved, to estimate the cost of the infrastructure under analysis. The results obtained from the dependability models and cost equations are then used in the next step to assist in the decision-making process.

Decision making (iv): At this stage, the cloud infrastructures are ranked based on any of the following distance measures: Euclidean, Manhattan and Minkowski. First, it is defined a criteria (e.g.: availability or cost) and objectives which can be minimize or maximize the criteria previously defined. The weights of each criterion according to the decision maker's preference are then defined. Lastly, a set of alternative solutions is ranked, based on a distance measure chosen. If the results are satisfactory for the desired criteria, the proposed strategy is complete. Otherwise, adjustments are made in the criteria, and the macro activity steps are repeated. Note that this macro step is automated, and the tool developed for this is described in "[MiPACE: a multi-criteria tool for planning and analysis of cloud environments](#)" section.

### The cloud environment

The base cloud architecture used for this study is depicted in Fig. 2, and comprises three main components: the main node, standby node, and the front-end. The main node consists of a virtual machine (VM) hosted on physical hardware (Hw). The virtual machine is represented by an operating system (OS) and an application service (APP). The application running in the VM is a digital library service. It should be noted, however, that the hardware in the main node supports an OS, a management server (Mng) and a VM. The management server executes the cloud services in the operating system. The standby node is used to ensure high levels of availability, and it assumes the role of the main node when a failure occurs; this node has the same components as the main node. The front-end is responsible for supervising and controlling the entire cloud environment through a specific cloud management tool. It is important to highlight that the remote storage volume can be accessed by the VMs, and is managed through the front-end. All of the components are interconnected by a private network. Note that from the base cloud architecture, more complex scenarios were considered based on the strategy described above and are described in the results and discussion section.

The cloud operational mode is described as follows. The main node (and its VM) and the front-end must be in working order for the system to be operational. However, if the standby and main nodes fail, the cloud becomes unavailable. The roles of the standby and main nodes are swapped when the VM is restored. The objective of the



**Fig. 2** Infrastructure overview

standby node is to maximize the availability of the cloud infrastructure, which can be established through a Service Level Agreement (SLA).

### Hierarchical models and cost equations

This section describes the hierarchical models designed to represent the base cloud environment previously presented (see Fig. 2). RBDs are used to represent the dependability relationship between independent subsystems, while detailed or more complex fail and repair mechanisms are modeled using SPNs. This approach enables the representation of many kinds of dependency between components, and avoids the well-known issue of state-space explosion [43]. Furthermore, this section also presents the proposed equations for estimating the cloud environment costs, which consider associated maintenance and operational costs.

#### Availability models for the base cloud environment

A hierarchical model was created to compute dependability-related metrics for the cloud environment described in “The cloud environment” section. Assuming cloud environments only, the architecture can be divided into three sub-models: front-end, main node, and standby node. The base cloud environment illustrated in Fig. 2 is modeled through RBDs and the respectively availability (reliability) is shown as:

$$P_s = P_{PE} \times (1 - (1 - P_{mn})(1 - P_{sn})), \quad (5)$$

where  $P_{PE}$ ,  $P_{mn}$  and  $P_{sn}$  is the front-end, main node and standby node availability (reliability), respectively.

Equation 6 computes the availability for the front-end sub-model, which is composed of three components connected in series: hardware, operating system, and management server. The front-end component is responsible for identifying and managing the underlying virtualized resources (i.e., the servers, network, and storage). The hardware component corresponds to the physical parts of a computer system (i.e., the memory, CPU, network, etc.). The cloud OS primarily manages the operation of one or more virtual machines within a virtualized environment, while the management server executes the cloud services in the operating system.

$$P_s = P_{Hw} \times P_{OS} \times P_{Mg} \quad (6)$$

where  $P_{Hw}$ ,  $P_{OS}$  and  $P_{Mg}$  is the hardware, operating system, and management server availability (reliability), respectively.

Equation 7 computes the availability for the main node. This node represents the computer resources for the deployment of virtual machines, and is composed of five components in series: hardware, operating system, management server, virtual machine, and service. Similar to the main node, the standby node is composed of five

components in series: hardware, operating system, management server, virtual machine, and service. We assumed that the components of the standby node have the same dependability characteristics as the main node; i.e., the same MTTFs and MTTRs.

$$P_s = P_{Hw} \times P_{OS} \times P_{Mg} \times P_{Vm} \times P_{Sv} \quad (7)$$

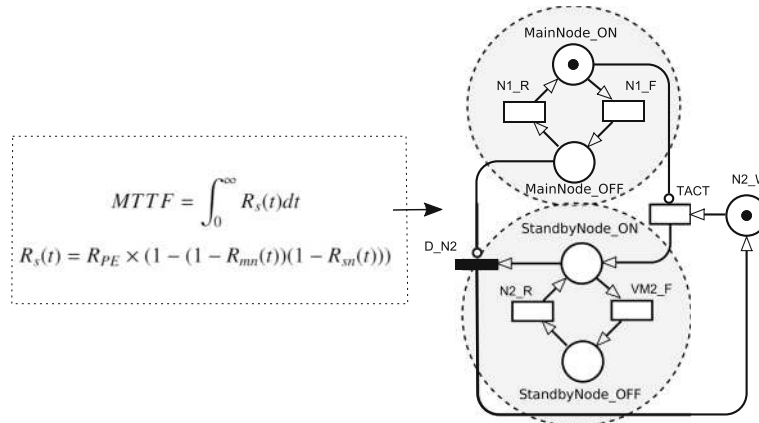
where  $P_{Hw}$ ,  $P_{OS}$  and  $P_{Mg}$  is the hardware, operating system, management server, virtual machine, and service availability (reliability), respectively.

The availability model representing the base cloud environment depicted in Eq. 5 operates in a hot-standby redundancy configuration (indicated by the parallel configuration). That is, when the main node fails, the redundant component replaces it without a delay in activation. This type of redundancy improves the system availability, because when the main node fails, the hot-standby node automatically takes its place. Nevertheless, RBDs equations cannot easily handle detailed failure/repair behavior. The warm-standby and cold-standby replication mechanisms cannot be fully represented in RBD models, due to the dependency between states of components. Therefore, such mechanisms are represented by SPNs. More specifically, in this paper the warm-standby and cold-standby replication mechanisms are adopted for the main node and virtual machines components; Fig. 3 presents an example of an SPN model for a node with cold-standby redundancy. Note that the hierarchical model solution is computed by passing the outputs of lower-level sub-models as inputs to the higher level sub-models. For example, the results from an SPN model representing a redundant VM are passed as values to the base cloud environment model. The base model is then solved to compute dependability metrics.

#### Cold standby model

A component with cold standby redundancy is based on a nonactive spare module that waits to be activated when the (main) active module fails. Hence, when the main module fails, the spare module's activation takes a certain amount of time to be activated. This time period is named mean time to activate (TACT). As the spare component is switched off, it is considered that it does not fail until becoming operational.

Figure 4 depicts an SPN model that illustrates this mechanism. The model uses two virtual machines in four possible places: VM1\_ON, VM1\_OFF, VM2\_ON, and VM2\_OFF. The places represent the operational and failure states for both main and spare modules. The spare module (VM2) is initially deactivated, so no tokens are stored in places VM2\_ON and VM2\_OFF. When the main module fails (VM1), the transition TACT is fired, and consequently the spare module is activated. The immediate transition D\_VM2 represents the deactivation of the spare



**Fig. 3** An illustrative example of a SPN model for a node with a cold-standby redundancy

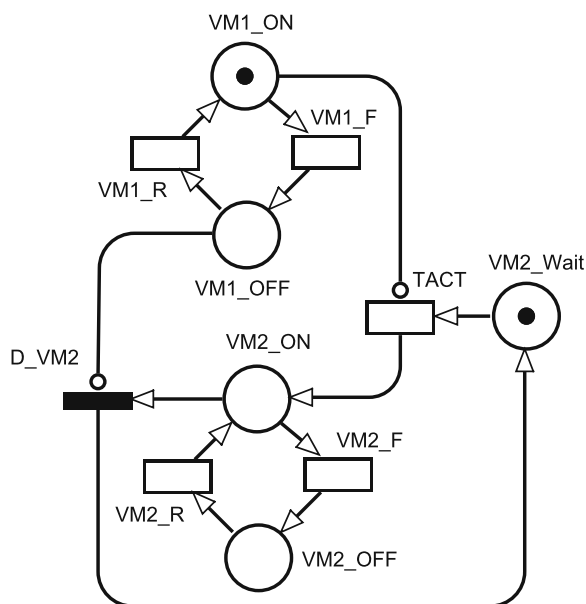
module when the main module is recovered. This redundancy mechanism fails if both modules fail. Thus, the operating mode can be expressed as

$$Cold_{Operational} = (VM1\_ON=1 \text{ OR } VM2\_ON=1) \quad (8)$$

where a token in the places VM1\_ON or VM2\_ON, defines the operational state of the environment.

#### Warm standby model

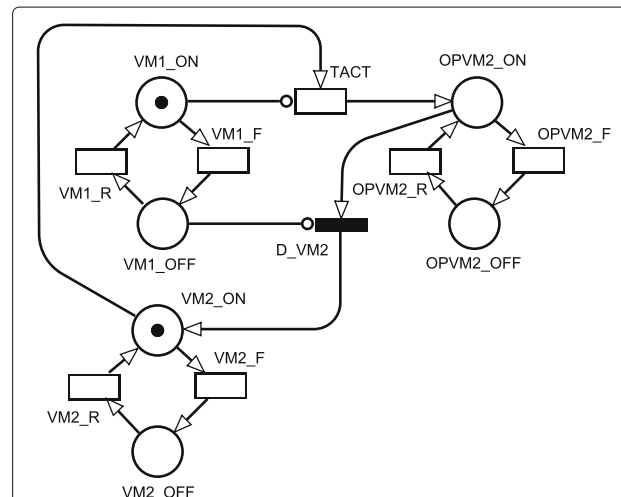
A component with warm standby redundancy is based on a nonactive spare module that waits to be activated when the active module fails. The difference with the cold standby redundancy is that the active and spare modules have failure rates  $\lambda$  and spare module has a failure rate  $\phi$  when it is de-energized, considering  $0 \leq \phi \leq \lambda$ .



**Fig. 4** SPN for cold-standby

Figure 5 illustrates an SPN warm standby model. The warm standby model has an active module with a full failure rate  $\lambda_F$  ( $1/MTTF\_VM1$ ), and the standby is operating with a reduced failure rate  $\alpha_F$  ( $1/MTTF\_OPVM1$ ). This redundancy mechanism has the spare module configured, but unavailable; it also ensures that the environment has continuously mirrored data. The spare module is activated in the presence of a fault in the environment, and consequently the time before activation will be shorter than in the cold standby approach.

When the main module fails, the secondary module is fully activated and replaces the faulty main component. The transition TACT represents the activation event. Places VM1\_ON, and VM1\_OFF represent the operational and non-operational states of the main module. Places OPVM2\_ON and OPVM2\_OFF represent the spare module in the operational state when not available. Places VM2\_ON and VM2\_OFF represent the situation in



**Fig. 5** SPN for warm-standby

which the secondary module fails before being activated, because it is regularly synchronized with the main module. When the main module fails, the transition TACT is fired to activate the spare module, similarly to the cold redundancy. The immediate transition is named D\_VM2, and has the same behavior in cold standby. The entire model fails if both modules fail. Thus, the operating mode can be expressed as

$$Warm_{Operational} = (VM1\_ON=1 \text{ AND } OPVM2\_ON=1) \quad (9)$$

where a token in places VM1\_ON or OPVM2\_ON represents the operational state of the environment.

Figure 6 depicts the SPN active-active (A/A) redundancy model. From this model, it is possible to estimate the capacity-oriented availability considering the service running on a set of VMs that are hosted on a node. The NVM and NND parameters allow such representation, where,  $n > 1$ . The places VM\_ON, ND\_ON, VM\_OFF, and ND\_OFF represent the operational and failure states for both VMs and Nodes. The transition DE is activated when there are no tokens in place ND\_ON, that is, when all nodes

fail. Thus, VMs will be failing if they fail, or when all nodes fail. The VM\_DW place represents the failure state of the VMs when all nodes are faulted. The RVM transition represents the return of the VMs to the operational state since the nodes have been repaired.

Equation 10 presents the COA calculation for active-active model considering a scenario with two virtual machines and one physical node, i.e.,  $NVM = 2$  and  $NND = 1$ .

$$COA = ((P\{\#VM1\_ON\} = (1 \times NVM)) \times (1 \times NVM)) + (P\{\#VM1\_ON\} = ((1 \times NVM) - 1)) \times ((1 \times NVM) - 1)) / (1 \times NVM) \quad (10)$$

### Cost model

The cost model uses the concept of Total Cost of Ownership (TCO) for evaluating and comparing the costs of cloud computing environments. TCO is the process of identifying costs categories other than price, transport, and operational [29, 46]. From the details of each experiment described above (such as the number of nodes, and service availability and unavailability), we allocate a period of time in which to estimate the total cost of each cloud environment under study. The estimate includes the cost of maintenance, operation, and rent of the cloud environment. Equation 11 estimates the total cost of the infrastructure.

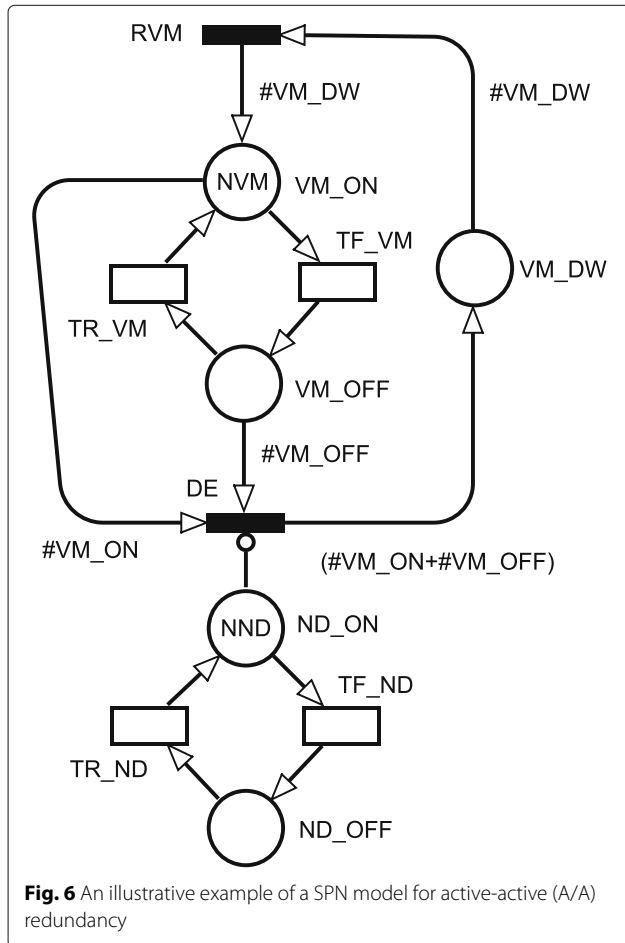
$$TCe = Cr + Cm + Cop \quad (11)$$

$Cr$ , which is represented by Eq. 12, allows determination of the costs associated with the rent of the cloud infrastructure.  $\sum Lc$  represents the monetary value paid for the components that make up the infrastructure, that is, the amount of investment made in equipment and facilities to keep the infrastructure in operation.  $N$  is the number of nodes deployed,  $T$  is the assumed time period, and  $Av$  is the availability of the infrastructure as a service.

$$Cr = \sum Lc \times N \times T \times Av \quad (12)$$

Equation 13 is used to estimate the maintenance costs (represented by  $Cm$ ).  $Dwt$  is the downtime period.  $Lb_{Dw}$  represents the maintenance labor cost per hour when a failure occurs.  $Sf$  is a service factor; i.e., customers may pay more or less depending on the contracted service, which affect the priority level for problem resolution.  $N$  and  $VM$  represent the number of nodes and virtual machines allocated by the contract, respectively.  $T$  is the period of service specified in the contract, while  $\sum Cr$  represents the costs related to the replacement of cloud components.

$$Cm = (Dwt \times Lb_{Dw} \times Sf \times N \times VM \times T) + \sum Cr \quad (13)$$



**Fig. 6** An illustrative example of a SPN model for active-active (A/A) redundancy

Equation 14 represents  $Cop$ , and allows the calculation of the operational costs of the cloud environment.  $Ec$  is the energy consumption, and  $E_p$  is the electricity price.  $Lb_{Up}$  represents the monetary value of each hour spent on keeping the infrastructure operational, while  $T$ ,  $Sf$ ,  $Av$ ,  $N$ , and  $VM$  represent the same parameters as presented in the previous equations.

$$Cop = (Ec \times E_p \times N \times T \times Av) + (Lb_{Up} \times Sf \times Av \times N \times VM \times T) \quad (14)$$

### MiPACE: a multi-criteria tool for planning and analysis of cloud environments

This section is dedicated to presenting the details of the developed tool. MiPACE was developed to support the planning of cloud infrastructures which consider customer service constraints, and tool assists in the decision-making process. It allows analysts, technicians, managers, and users of cloud services to plan and analyze cloud scenarios. The tool is written in the programming language C, and the features implemented are described below.

- (i) Mercury tool: The Mercury tool [40] was developed by the MODCS research group, and allows the creation and evaluation of performance and dependability models. It implements the following formalisms: Continuous Time Markov Chains (CTMCs), Reliability Block Diagrams (RBDs), Energy Flow Models (EFMs), and Stochastic Petri nets (SPNs). The Mercury tool is used along with the MiPACE tool to create hierarchical models, and to solve the experimental study design scenarios.
- (ii) Integration module: Because MiPACE does not implement the RBD and SPN formalisms, this module was implemented to integrate the results obtained from the Mercury tool into MiPACE. That is, an input file is created with all of the results obtained from the design of experiment studies, and then uploaded into the MiPACE tool.
- (iii) Design of experiment editor: This feature allows users to plan experiments. Initially, it is necessary to choose a number of factors to be combined. The user then indicates the number of levels for each factor. Note that the tool supports the full factorial method, which involves testing every combination of factors against each other. As explained earlier, the experiment design is adopted to investigate the effects of variations of one or more parameters in the base cloud environment; we therefore generate distinct SPN models from the SPN model that represents the base cloud environment, and investigate the impact of such modifications on the adopted metrics. Thus, the purpose of this feature is

to provide a set of scenarios that will be modeled and analyzed by the Mercury tool.

- (iv) Ranking generator: When the results obtained from the experiment study designs have been uploaded into MiPACE, this tool then ranks a set of optimal solutions. At this stage, the user of the tool must define the criteria function (e.g. availability or cost) and the objective which be minimized or maximized the criteria previously defined. The user can then choose the distance measure for ranking the solutions, such as the Euclidean, Manhattan or Minkowski distances [16]. These distance measures are used for similarity comparisons. Finally, the user can add weights to the criteria function previously defined to prioritize one variable over another; for example, the user could use this option to prioritize cost over high availability.
- (v) Report tool: The results that consider each criterion are displayed in the tool panel. MiPACE also generates two output files containing the ranking of the architectures, with one output file used for visualization and the other for plotting purposes. If necessary, the user can change the criterion function or objective, and repeat the ranking step.

## Results and discussion

This section discusses a case study to illustrate the applicability of the proposed approach when considering availability, capacity-oriented availability, reliability and cost requirements. The approach assists individuals in identifying an ideal cloud infrastructure, and takes service constraints into account. The availability and cost models are useful during the design and analysis of cloud infrastructures, because they represent the characteristics of cloud environments. The results obtained by the evaluation of these models serve as the input to the MiPACE tool, which then finds a set of optimal solutions.

### Evaluation of the base cloud environment

The first part of this case study aims to demonstrate the applicability of the availability models, and presents the results obtained for the base cloud environment. The base cloud environment represented in Fig. 2 was modeled (shown in Eqs. 6 and 7), and the availability models combined to represent the whole cloud infrastructure. Equation 5 illustrates the RBD model for the base cloud environment. The reliability importance index<sup>1</sup> (RI) was the adopted to identify which component of the system required further attention to increase the availability level. Assuming only the Front-End and Main nodes of the devices present in Fig. 2, the RI index for the nodes was, 0.153201 and 0.219544, respectively. The main node is the most critical component, and it is most important when adopting a redundancy mechanism. Three redundancy

**Table 1** Cost parameters

Parameter	Value
$E_p$	0.1547 (USD)
$E_c$	0.4 (kWh)
$Lb_{Up}$	0.04 (USD)
$Lb_{Dw}$	0.40 (USD)
<i>Gold</i>	1.42
<i>Silver</i>	1.26
<i>Bronze</i>	1.15

strategies hot, cold, and warm were used to increase the availability levels, and these mechanisms are presented in Eqs. 4 and 5, respectively.

Table 1 shows the parameter values adopted for estimating the cost of the cloud environment. The  $E_p$  and  $E_c$  parameters represent the energy price (in USD) and the energy consumption per kilowatt-hour [13], respectively. Such parameters only take the servers into account. The  $Lb_{Up}$  (operation) and  $Lb_{Dw}$  (maintenance) parameters indicate the labor cost per hour, while  $R_t$  represents the rental rate for the cloud infrastructure. The type of service is categorized as *gold*, *silver*, or *bronze*, and these reflect the capacity of the cloud maintenance team to support different quality levels; a reduction of 10% in the mean time to repair the silver service in comparison to the gold service assumed, with a reduction of 20% in the mean time to repair the bronze service in comparison to the gold service.

Table 2 presents the Mean Time to Failure (MTTF) and Mean Time to Repair (MTTR) used for the availability model (Eq. 5). Those values were obtained from [2, 8, 20], and are used to compute dependability-related metrics for the sub-models, and then for the whole system.

Table 3 shows the parameter values used for the sub-model (Eq. 6), based on [2, 8, 20].

Table 4 shows the input parameters for the cold-standby SPN model based on [2, 8, 20]. The parameter values used for evaluating the model may be modified to represent, for example, different service repair policies. It is thus possible to analyze situations where the firing of transitions is shorter or longer depending on the adopted repair policy.

**Table 2** Parameters for the front-end submodel

Component	Value (hr)
MTTF_Hw-front	8760
MTTR_Hw-front	1.67
MTTF_OS-front	1440
MTTR_OS-front	1
MTTF_Mng-front	788.4
MTTR_Mng-front	1

**Table 3** Parameters for the cloud node sub-model

Component	Value (hr)
MTTF_Hw-node	8760
MTTR_Hw-node	1
MTTF_OS-node	1440
MTTR_OS-node	1
MTTF_Mng-node	788.4
MTTR_Mng-node	1
MTTF_VM-node	2880
MTTR_VM-node	0.5
MTTF_Service	6865.3
MTTR_Service	0.17

Furthermore, values regarding the mean time to failure or mean time to repair may represent components with higher or lower reliability. These models can assist individuals in identifying service repair policies that fit their needs.

Table 5 shows the input parameters for the *warm-standby* model. Like the *cold-standby* model, the values relating to the mean time to repair and mean time to failure can be modified in order to represent, for example, more reliable repair policies. Such modifications help individuals in planning cloud environments that fit their needs and budgets.

Table 6 presents the evaluation of the base cloud environment under distinct configurations. The first configuration is composed of the front-end and the main node, while the second comprises the front-end, the main node and a hot-standby redundancy of the main node. The third configuration includes the front-end, the main node and a cold-standby redundancy of the main node. Lastly, the fourth configuration is composed of the front-end, the main node, and warm-standby redundancy of the main node. After evaluation of the models that represent each of these configurations, it is possible to note the differences between each configuration in terms of availability, capacity-oriented availability, reliability and cost. Furthermore, the importance of redundant mechanisms is illustrated by comparing the first configuration to configurations (2, 3 and 4).

**Table 4** Parameters for the cold-standby SPN model

Transition name	Value (hr)
MTTF_VM1	2028
MTTR_VM1	0.40
MTTF_VM2	2028
MTTR_VM2	0.40
D_VM2	-
TACT	0.35

**Table 5** SPN Warm-Standby parameters

Transition name	Value (hr)
MTTF_VM1	2028
MTTR_VM1	0.40
MTTF_VM2	2028
MTTR_VM2	0.40
MTTF_OPVM1	2434
MTTR_OPVM1	0.40
TACT	0.1667
D_VM2	-

The downtime for each configuration was also calculated. We considered the downtime in minutes over a period of one month, and the following values were obtained: (1) 193.99 min, (2) 93.05 min, (3) 120.88 min, and (4) 108.60 min. As expected, the hot-standby mechanisms had the lowest downtime in relation to the other configurations, followed by warm, cold and finally the configuration without redundancy.

The third column of Table 6 describes the values obtained for the reliability analysis of each configuration. The reliability metric was obtained through transient analysis over a time (T) period of 24 hours. The reliability analysis assumes that the system cannot be repaired. The last column of Table 6 shows the estimated total cost (TCe) for each configuration; as expected, configuration 1 had the lowest TCe because it has the simplest configuration. In the next subsection, a *n* experiment study design is performed with the base cloud environment, in order to generate a set of scenarios that will be ranked using an MCDM method.

### Planning for design of experiments

From the base cloud environment, we designed an experiment to identify which of the variables have the greatest influence on the adopted metrics (i.e., capacity oriented-availability, availability or cost). Table 7 illustrates the experimental plan that considers the variables from Table 8. We generated 72 configurations, where each case received sequential numbering (1 to 72). However, we have removed 36 because some of these configurations cannot be represented in the availability models. This can happen due to the full factorial method adopted.

**Table 6** Results obtained by solving the models for the base cloud environment

Configuration	Av. (%)	COA (%)	Rel. (%)	TCe (USD)
Conf. (1)	99.55095747	99.76543388	89.447	306.31
Conf. (2)	99.78461098	99.97929139	94.797	348.84
Conf. (3)	99.72018536	99.92583000	94.888	338.83
Conf. (4)	99.74862082	99.94998263	94.788	331.84

**Table 7** Planning for design of experiments of the first and second scenarios

Conf.	Node	VM	TS	RT	Conf.	Node	VM	TS	RT
1	1	1	Gold	N/R	34	2	2	Bronze	Hot
5	1	1	Silver	N/R	35	2	2	Bronze	Cold
9	1	1	Bronze	N/R	36	2	2	Bronze	Warm
14	1	2	Gold	Hot	38	2	4	Gold	Hot
15	1	2	Gold	Cold	39	2	4	Gold	Cold
16	1	2	Gold	Warm	40	2	4	Gold	Warm
18	1	2	Silver	Hot	42	2	4	Silver	Hot
19	1	2	Silver	Cold	43	2	4	Silver	Cold
20	1	2	Silver	Warm	44	2	4	Silver	Warm
22	1	2	Bronze	Hot	46	2	4	Bronze	Hot
23	1	2	Bronze	Cold	47	2	4	Bronze	Cold
24	1	2	Bronze	Warm	48	2	4	Bronze	Warm
26	2	2	Gold	Hot	50	3	3	Gold	Hot
27	2	2	Gold	Cold	54	3	3	Silver	Hot
28	2	2	Gold	Warm	58	3	3	Bronze	Hot
30	2	2	Silver	Hot	62	3	6	Gold	Hot
31	2	2	Silver	Cold	66	3	6	Silver	Hot
32	2	2	Silver	Warm	70	3	6	Bronze	Hot

This method involves testing every combination of factors against each other, and some combinations cannot be applied in a real cloud environment. Service providers or individuals that adopt this approach, must check the experimental plan to identify any inconsistencies. For example, if the experimental planning generates a scenario where the number of nodes and the number of VMs are equal to 1, the redundancy mechanism factor should only be assigned to the *no redundancy* (N/R) level (see Table 8). This is because modelling a configuration with redundancy requires the number of nodes or VMs to be greater than 1. Thus, we kept the numbering initially generated after the removal of several scenarios that could not be represented.

Table 8 presents an overview of the factors (i.e., variables, (*k*)) and the levels (*n<sub>i</sub>*) applied in the design of the experiments. We considered two levels of redundancy for the node factor, and considered up to three levels for the VM factor. The type of service (TS) factor represents the

**Table 8** Factors and levels

Factors	Levels			
Nodes	1	2	-	-
VMs per node	1	2	3	-
Type of service	Gold	Silver	Bronze	-
Redundancy Type	N/R	Hot	Cold	Warm

maintenance factors adopted in this paper, and reflects the capacity of the cloud maintenance team to support different levels of maintenance quality. In this sense, we consider a reduction of 10% in the mean time to repair using a *silver* service compared to a *gold* service, and a reduction of 20% in the mean time to repair using a *bronze* service compared to a *gold* service.

The redundancy type (RT) factor refers to the provision of support for the redundancy feature. *Hot* redundancy

is commonly used when the system must not go down, even briefly, under any condition. This mechanism uses a spare component in the same regime as the primary one, and the redundant unit is fully capable of supporting the primary unit. *Cold* redundancy switches to the reserve unit only after failure of the primary unit. For the switch to take place, a time is scheduled for the substitution of the primary module for the reserve module. *Warm* redundancy tends to decrease the switching time for the reserve

**Table 9** Dependability and cost results for each configuration

Conf.	Availability (%)	COA (%)	Reliability (%)	Downtime (Min)	TCe (USD)
1	99.5509574705	99.7654338777	89.4469247967	196.68062793	306.315
5	99.5048180576	99.7398556502	89.4469247967	216.88969077	301.106
9	99.4587108348	99.7142905588	89.4469247967	237.08465436	297.616
14	99.5706543903	99.8728535266	90.2149434700	188.05337705	306.292
15	99.5616066096	99.7759681848	90.4897545045	192.01630500	306.302
16	99.5650001940	99.7793663489	90.4925359715	190.52991502	306.299
18	99.5280378157	99.8576135555	90.2149434700	206.71943671	301.090
19	99.5194114138	99.7543530012	90.4897545045	210.49780078	301.096
20	99.5233822554	99.7583295360	90.4925359715	208.75857214	301.093
22	99.4854486046	99.8423824134	90.2149434700	225.37351118	297.605
23	99.4774397512	99.7329443057	90.4897545045	228.88138896	297.608
24	99.4818832542	99.7373945400	90.4925359715	226.93513466	297.607
26	99.7846109835	99.9792913946	94.7970536818	94.34038921	348.837
27	99.7201853569	99.9258299977	94.8881075626	122.55881369	338.826
28	99.7486208259	99.9499826254	94.7885068480	110.10407827	331.844
30	99.7638083387	99.9741786568	97.970536818	103.45194767	348.939
31	99.6976733870	99.9196490290	94.8881075626	132.41905652	349.264
32	99.7272945753	99.9445963495	94.7885068480	119.44497603	349.119
34	99.7430013430	99.9690494692	94.7970536818	112.56541175	338.735
35	99.6754125418	99.9136202473	94.8881075626	142.16930671	339.006
36	99.7060521418	99.9392534800	94.7885068480	128.74916191	338.883
38	99.7846995558	99.9763972167	94.8827293630	94.30159457	331.721
39	99.7282517140	99.9363941560	94.9415323833	119.02574928	331.913
40	99.7534014438	99.9641014864	94.8717737486	108.01016760	331.828
42	99.7639238067	99.9712822680	94.8827293630	103.40137265	612.144
43	99.7061245286	99.9341938729	94.9415323833	128.71745648	612.275
44	99.7322176473	99.9633095757	94.8717737486	117.28867048	612.216
46	99.7431470289	99.9661503994	94.8827293630	112.50160136	601.884
47	99.6841904563	99.9323432327	94.9415323833	138.32458013	601.965
48	99.7110986022	99.9626569150	92.7019871643	126.53881225	601.928
50	99.7851593857	99.9802082869	95.1170633317	94.10018905	594.981
54	99.7644824363	99.9752797697	95.1170633317	103.15669289	594.997
58	99.7438139525	99.9703513633	95.1170633317	112.20948879	595.012
62	99.7851596846	99.9802044071	95.1242439537	94.10005812	612.095
66	99.7644828673	99.9752736682	95.1242439537	103.15650413	601.855
70	99.7438147914	99.9703425762	95.1242439537	112.20912137	595.012

module. Lastly, the N/R level (present in Table 8) means that the environment does not consider any redundancy approach.

When the design of experiments and the modeling of each scenario were complete, the next step was to assess the hierarchical models based on the RBDs and SPNs. We considered the following dependability metrics: availability, reliability and unavailability. Table 9 presents the results obtained for each scenario. For the analysis of downtime, we considered a time interval of one month (720 h), and the reliability was obtained by transient analysis over 24 h. The results for each scenario are described in Table 9. This table serves as input for our proposed approach, in that we present two case scenarios examining distinct customer constraints.

### First case scenario

To demonstrate the applicability of the multi-criteria approach, the first case considers a situation in which a given cloud user wishes to select a cloud environment with: lower cost and higher capacity-oriented availability. The distance measure selected for this case was the Euclidean distance. Table 10 illustrates the ranking of cloud environments considering the criteria and distance measure selected. Note that this ranking was automatically computed by the MiPACE tool (see “MiPACE: a multi-criteria tool for planning and analysis of cloud environments” section).

**Table 10** Configurations ranking for the defined criteria (i.e. minimize cost and maximize coa)

Ranking	Configuration	Ranking	Configuration
1	38	19	20
2	40	20	19
3	34	21	5
4	28	22	24
5	26	23	23
6	30	24	50
7	39	25	54
8	36	26	58
9	32	27	70
10	27	28	66
11	31	29	46
12	35	30	48
13	14	31	47
14	18	32	62
15	22	33	9
16	16	34	42
17	15	35	44
18	1	36	43

As Table 10, cloud configuration 38 is the best option for the defined criteria. This environment has the following configuration: two nodes and two VMs with hot-standby redundancy, and a gold service type. The second-best option in the ranking is configuration 40, which showed an increase in cost (0.03%) and decrease in COA (0.69%). The third-best option (configuration 34) had an increase in cost of 2.11% and an decrease in COA of 0.007% when compared to the first option. The worse scenario in the ranking (environment 43) had a increase in cost of 84.57% and in COA of 0.04% when compared to the best option. Table 11 illustrates the components adopted for each cloud configuration.

Figure 7 summarizes the configuration ranking generated for the first case scenario. Figure 7a gives an overview of all configurations described in Table 10, while Fig. 7b shows a set of optimal configurations. The y-axis represents the COA, while the x-axis represents the cost of the cloud configurations. The optimal set of results are plotted near to the x and y-axis, in accordance with the defined criteria (i.e., higher COA and lower cost). Figure 7b presents the optimal set of configurations at higher level of detail, and shows that configuration 38 is optimal.

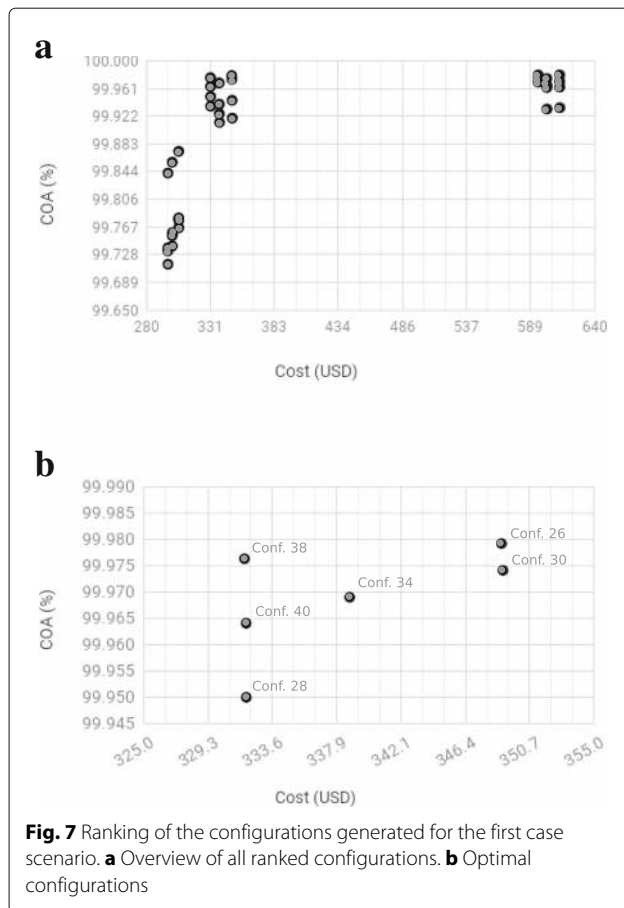
### Second case scenario

The second case scenario considers a situation in which a company or service provider wish to choose a cloud environment with higher reliability and lower cost. In this context, the multi-criterion function aims to maximize the first goal and minimize the second. When the decision variables (reliability and cost) have been selected, the ranking is applied using the MiPACE tool. As in the first case scenario, the Euclidean distance method was adopted to find a set of optimal solutions.

Table 12 presents the ranked configurations when considering the defined criteria, while Table 13 describes the components of the first and last configurations. Configuration 39 is the best-ranked configuration, and has the following components: two nodes and two VMs with cold-standby redundancy, and the gold service type. The second configuration in the ranking is configuration 38, which shows a reduction in reliability of 0.062% and a

**Table 11** Summary of the components used to rank the configurations (First case)

Ranking	Configuration	Node	VM	TS	RT
1	38	2	4	Gold	Hot
2	40	2	4	Gold	Warm
3	34	2	2	Bronze	Hot
...	...	...	...	...	...
36	43	2	4	Silver	Cold



reduction in cost of 0.058% (USD 0.19), when compared to the best-ranked configuration. The third-best configuration (40) had a reduction in reliability and cost of 0.074% and 0.026% respectively, when compared to the best-ranked configuration. The worst configuration in the ranking is configuration 48, which shows a decrease in reliability of 2.416% and an increase in cost of 44.858% when compared to the best configuration. Despite the reduction in cost of the configurations ranked in second and third positions, their reliability also decreased due to the type of redundancy adopted.

Figure 8 summarizes the configurations rankings generated for the second case scenario. Figure 8a presents an overview of all configurations described in Table 12, while Fig. 8b presents the details of the optimal set of configurations. The y-axis indicates the reliability of the cloud environment and the x-axis represents the cost. The defined criteria of maximizing reliability and minimizing cost, therefore mean that the optimal set of configurations are shown by points plotted in the lower right side of the figure (Fig. 8a). Figure 8b shows that configuration 39 is optimal.

**Table 12** Configuration ranking for the defined criteria (i.e. maximize reliability and minimize cost)

Ranking	Configuration	Ranking	Configuration
1	39	19	22
2	38	20	18
3	40	21	14
4	28	22	50
5	27	23	54
6	35	24	70
7	34	25	58
8	36	26	66
9	31	27	47
10	26	28	46
11	30	29	62
12	32	30	9
13	24	31	5
14	20	32	1
15	16	33	42
16	23	34	43
17	19	35	44
18	15	36	48

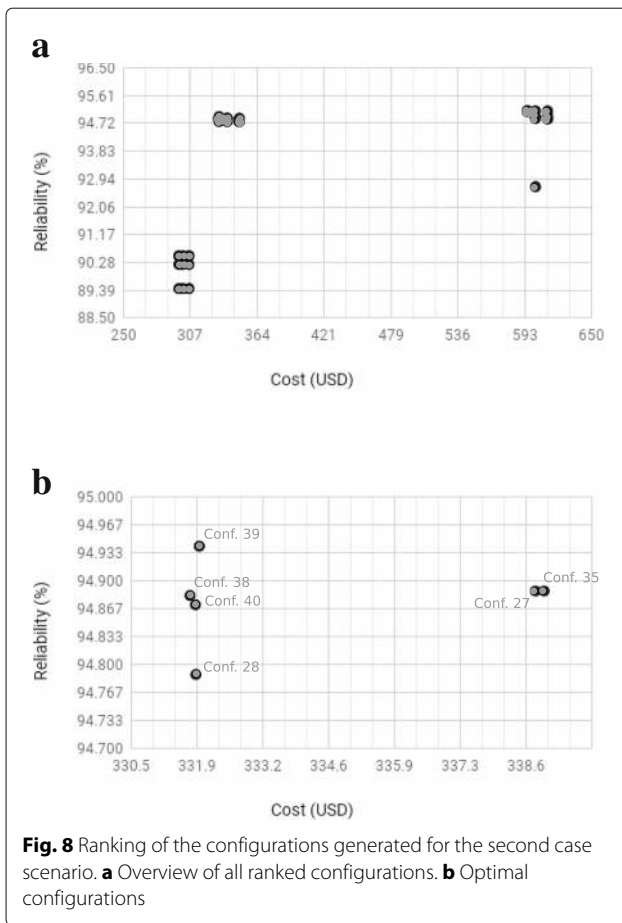
### Third case scenario

The third case scenario considers a situation in which a company or service provider wishes to choose a cloud environment with higher COA and lower cost. In order to contemplate this case, we adopt the active-active (A/A) redundancy model and consequently generate another design of experiments. Table 14 illustrates the factor values and levels and Table 15 presents an overview of the factors (i.e., variables, ( $k$ )) and the levels ( $n_i$ ) applied in the design of the experiments.

Table 16 presents the ranked configurations when considering the defined criteria, while Table 17 describes the components of the first and last configurations. Configuration 31 is the best-ranked configuration and has the following components: eight nodes and sixteen VMs with active-active redundancy, and the gold service type. The second configuration in the ranking is configuration 1,

**Table 13** Summary of the components used to rank the configurations (Second Case)

Ranking	Configuration	Node	VM	TS	RT
1	39	2	4	Gold	Cold
2	38	2	4	Gold	Hot
3	40	2	4	Gold	Warm
...	...	...	...	...	...
36	48	2	4	Bronze	Warm



which shows a decrease in COA (0.002%) and decrease in cost (61.89%) when compared to the best-ranked configuration. The third-best configuration (4) had a decrease in COA of 0.002% and decreased in cost (41.26%) when compared to the best-ranked configuration. The worst configuration in the ranking is configuration 60, which shows an increase in the cost of 1229.35% when compared to the best configuration. In spite of the increase in the number of nodes and VMs, and consequently an increase in the cost of the classified configurations, we noticed that the COA remained close.

Figure 9 summarizes the configurations rankings generated for the third case scenario. Figure 9a presents an overview of all configurations described in Table 17, while

**Table 14** Factors and levels

Factors	Levels				
Nodes	2	4	8	16	-
VMs per node	2	4	8	16	32
Type of service	Gold	Silver	Bronze	-	-
Redundancy type	A/A	-	-	-	-

**Table 15** Planning for design of experiments of the third scenario

Conf.	Node	VM	TS	RT	Conf.	Node	VM	TS	RT
1	2	4	Gold	A/A	31	8	16	Gold	A/A
2	2	4	Silver	A/A	32	8	16	Silver	A/A
3	2	4	Bronze	A/A	33	8	16	bronze	A/A
4	2	8	Gold	A/A	34	8	32	Gold	A/A
5	2	8	Silver	A/A	35	8	32	Silver	A/A
6	2	8	Bronze	A/A	36	8	32	Bronze	A/A
7	2	16	Gold	A/A	37	8	64	Gold	A/A
8	2	16	Silver	A/A	38	8	64	Silver	A/A
9	2	16	Bronze	A/A	39	8	64	Bronze	A/A
10	2	32	Gold	A/A	40	8	128	Gold	A/A
11	2	32	Silver	A/A	41	8	128	Silver	A/A
12	2	32	Bronze	A/A	42	8	128	Bronze	A/A
13	2	64	Gold	A/A	43	8	256	Gold	A/A
14	2	64	Silver	A/A	44	8	256	Silver	A/A
15	2	64	Bronze	A/A	45	8	256	Bronze	A/A
16	4	8	Gold	A/A	46	16	32	Gold	A/A
17	4	8	Silver	A/A	47	16	32	Silver	A/A
18	4	8	Bronze	A/A	48	16	32	Bronze	A/A
19	4	16	Gold	A/A	49	16	64	Gold	A/A
20	4	16	Silver	A/A	50	16	64	Silver	A/A
21	4	16	Bronze	A/A	51	16	64	Bronze	A/A
22	4	32	Gold	A/A	52	16	128	Gold	A/A
23	4	32	Silver	A/A	53	16	128	Silver	A/A
24	4	32	Bronze	A/A	54	16	128	Bronze	A/A
25	4	64	Gold	A/A	55	16	256	Gold	A/A
26	4	64	Silver	A/A	56	16	256	Silver	A/A
27	4	64	Bronze	A/A	57	16	256	Bronze	A/A
28	4	128	Gold	A/A	58	16	512	Gold	A/A
29	4	128	Silver	A/A	59	16	512	Silver	A/A
30	4	128	Bronze	A/A	60	16	512	Bronze	A/A

Fig. 9b presents the details of the optimal set of configurations. The y-axis indicates the cost of the cloud environment and the x-axis represents the capacity-oriented availability (COA). The defined criteria of minimizing cost and maximizing COA, therefore mean that the optimal set of configurations are shown by points plotted in the lower right side of Fig. 9a. Figure 9b depicts that configuration 31 is optimal.

### Final remarks

In this paper, we presented an approach to model and analyze cloud infrastructures while considering availability, capacity-oriented availability (COA), reliability and cost requirements. This approach uses stochastic models and a multiple-criteria decision-making method to calculate

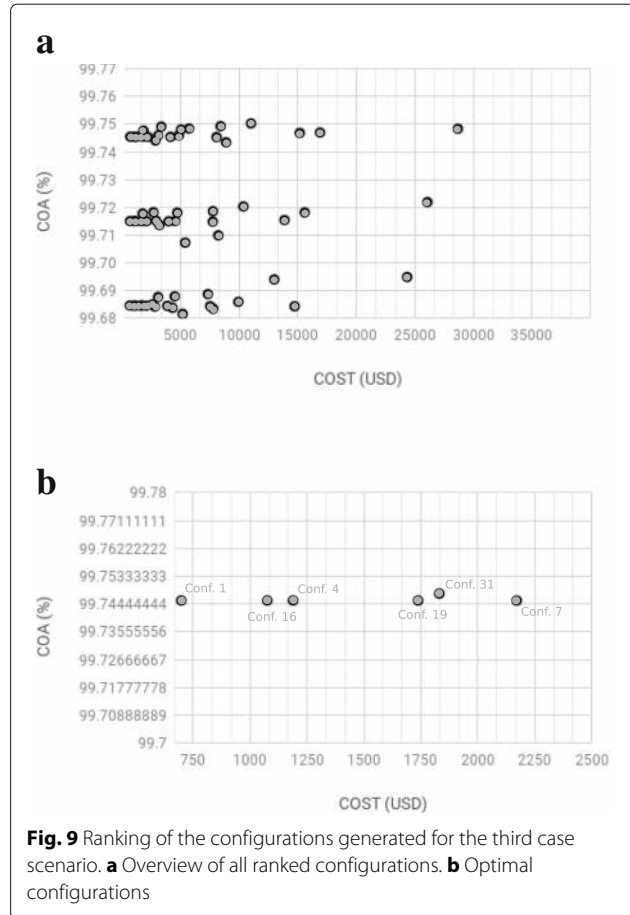
**Table 16** Configuration ranking for the defined criteria (i.e. maximize COA and minimize cost)

Ranking	Configuration	Ranking	Configuration
1	31	31	47
2	1	32	29
3	16	33	14
4	4	34	43
5	19	35	41
6	7	36	26
7	46	37	56
8	22	38	44
9	34	39	48
10	10	40	51
11	49	41	54
12	37	42	57
13	25	43	36
14	13	44	3
15	52	45	18
16	40	46	33
17	28	47	6
18	35	48	21
19	32	49	9
20	50	50	12
21	2	51	24
22	17	52	39
23	5	53	15
24	20	54	30
25	8	55	59
26	23	56	58
27	55	57	42
28	53	58	27
29	11	59	45
30	38	60	60

dependability-related metrics, and to rank cloud infrastructures. A hierarchical strategy was used to modeling and planning cloud infrastructures, combining a multiple-criteria decision-making to find the most appropriate from the set. A case study was presented to illustrate the feasibility of the proposed approach. The results show

**Table 17** Summary of the components used to rank the configurations (Third Case)

Ranking	Configuration	Node	VM	TS	RT
1	31	8	16	Gold	A/A
2	1	2	4	Gold	A/A
3	16	4	8	Gold	A/A
...	...	...	...	...	...
60	60	16	512	Bronze	A/A



that our approach enables service providers or individuals who are interested in building their own clouds, to choose the most appropriate cloud infrastructure; the approach allows multiple criteria such as reliability, downtime, and cost to be considered.

As future work, we plan to apply our approach in a more complex environment by utilizing other MCDM methods. We also consider evaluating other metrics like response time, throughput, and CPU usage.

## Endnote

<sup>1</sup> The index indicates the impact of a particular component in the overall system reliability [21].

## Abbreviations

A/A: Active-active; A: Availability; AHP: Analytic hierarchy process; ANP: Analytic network process; APP: Application service; COA: Capacity-oriented availability; CTMC: Time markov chain; ELECTRE III: Elimination and choice expressing reality; FT: Fault trees; Hw: Physical hardware; IaaS: Infrastructure as a service; MAGDM: multi-attribute group decision-making; MCDM: Multiple-criteria decision-making; MTTF: Mean time to failure; MTTR: Mean time to repair; N/R: No redundancy; OS: Operating system; PaaS: Platform as a service; QoS: Quality of service; RBD: Reliability block diagrams; RI: Reliability importance; RT: Redundancy type; SaaS: Software as a service; SLA: Service level agreement; SPNs: Stochastic petri nets; TCO: Total cost of ownership; TS: Type of service; TTF: Time to failure; TOE: Technology, organization, and

environment; TOPSIS: Technique for Order of Preference by Similarity to Ideal Solution; VM: Virtual machine

### Acknowledgements

The authors would like to thank the reviewers for their valuable comments and suggestions to improve the quality of this work.

### Funding

Not applicable.

### Availability of data and materials

All data and models used in the production of this research may be made available at any time by the publishers.

### Authors' contributions

The author and co-authors worked collaboratively on generating the results and writing the article. All authors read and approved the final manuscript.

### Authors' information

Julian Araujo is a Brazilian Ph.D. student in Computer Science at the Federal University of Pernambuco. He graduated in Industrial Automation from the CEFET-CE in 2005 and Electromechanical from the CENTEC-CE in 2005. Since 2011 he has been a member of the Department of Statistics and Informatics of Universidade Federal Rural de Pernambuco, where he is currently Assistant Professor. His research focuses on performance modeling, reliability and availability modeling, Stochastic Petri net and fault tolerant computing. Paulo Maciel received the degree in electronic engineering in 1987 and the M.Sc. and Ph.D. degrees in electronic engineering and computer science from the Federal University of Pernambuco, Recife, Brazil, respectively. He was a faculty member with the Department of Electrical Engineering, Pernambuco University, Recife, Brazil, from 1989 to 2003. Since 2001, he has been a member of the Informatics Center, Federal University of Pernambuco, where he is currently an Associate Professor. In 2011, during his sabbatical from the Federal University of Pernambuco, he stayed with the Department of Electrical and Computer Engineering, Edmund T. Pratt School of Engineering, Duke University, Durham, NC, USA, as a Visiting Professor. His current research interests include performance and dependability evaluation, Petri nets and formal models, encompassing manufacturing, embedded, computational, and communication systems as well as power consumption analysis. Dr. Maciel is a Research Member of the Brazilian Research Council.

Ermeson Andrade is an adjunct professor at the Department of Statistics and Informatics at Federal Rural University of Pernambuco (DEINFO-UFRPE), Brazil. From 2015 to 2016, he was a Postdoctoral Fellow at VU University Amsterdam, Netherlands. In March 2014, he finished his Ph.D degree in Computer Science at Informatics Center at Federal University of Pernambuco (Cin-UFPE). From 2010 to 2011, he was a visiting scholar at the Department of Electrical and Computer Engineering at Duke University, USA, under the supervision of Dr. Kishor Trivedi. He also has over ten years of experience in the industry, working with test and quality assurance.

Gustavo Callou is an Associate Professor at Federal Rural University of Pernambuco, Brazil. He holds a PhD in Computer Science from Federal University of Pernambuco, Brazil, with an enrolment period at Bergische Universität Wuppertal, Germany, in the area of Performance Evaluation. His main research interests are Petri nets, Dependability, Reliability Analysis, Fault Tolerant Computing, Performance Engineering, Sustainability, Computer Network, Cloud Computing and Data Centers, where, he has published over 45 papers on these topics in journals, international conferences, and workshops. Vandi Alves was born in Recife, Brazil. He received the B.Sc. degree in computer engineering in 2017 from Federal University of Pernambuco. His research interests include petri nets, embedded systems and performance evaluation. Paulo Cunha was born in Recife, Brazil. He received the B.Sc. degree in electrical engineering and the M.Sc. degree in computer science from the Federal University of Pernambuco (UFPE), Recife, Brazil, in 1974 and 1977, respectively, and the Ph.D. degree in computer science from the University of Waterloo, Waterloo, ON, Canada, in 1981. He is currently a Full Professor with the Center of Informatics, UFPE. His research interests include computer networks and distributed systems, formal methods, and distributed software architectures. He has published several papers in the areas of programming languages and methodologies, formal description techniques, and distributed systems.

### Competing interests

Not applicable.

### Publisher's Note

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

### Author details

<sup>1</sup>Federal University of Pernambuco (UFPE), Informatics Center (CIn), Recife, Brazil. <sup>2</sup>Federal Rural University of Pernambuco (UFRPE), Department of Statistics and Informatics (DEINFO) Recife, Brazil.

Received: 10 October 2017 Accepted: 1 March 2018

Published online: 27 March 2018

### References

- Andrade EC, Alves M, Matos R, Silva B, Maciel P (2013) Openmads: An open source tool for modeling and analysis of distributed systems. In: International Conference on Computer Safety, Reliability, and Security. Springer Berlin Heidelberg. pp 277–284
- Araujo J, Maciel P, Torquato M, Callou G, Andrade E (2014) Availability evaluation of digital library cloud services. In: Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on. IEEE. pp 666–671
- Bauer E, Adams R (2012) Reliability and availability of cloud computing. Wiley
- Bauer E, Adams R (2014) Service Availability Measurement, chap. 12. Wiley-IEEE Press. <https://doi.org/10.1002/9781118763407.ch12>
- Bauer E, Adams R, Eustace D (2011) Beyond Redundancy: How Geographic Redundancy Can Improve Service Availability and Reliability of Computer-based Systems. Wiley
- Bosse S, Splieth M, Turowski K (2016) Multi-objective optimization of {IT} service availability and costs. Reliab Eng Syst Saf 147:142–155. <https://doi.org/10.1016/j.res.2015.11.004>
- Dai W, Qiu L, Wu A, Qiu M (2017) Cloud infrastructure resource allocation for big data applications. IEEE Trans Big Data PP(99):1–1. <https://doi.org/10.1109/TBDA.2016.2597149>
- Dantas J, Matos R, Araujo J, Maciel P (2012) An availability model for eucalyptus platform: An analysis of warm-standby replication mechanism. In: Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on. pp 1664–1669. <https://doi.org/10.1109/ICSMC.2012.6377976>
- Dantas J, Matos R, Araujo J, Maciel P (2015) Eucalyptus-based private clouds: availability modeling and comparison to the cost of a public cloud. Computing 97(11):1121–1140. <https://doi.org/10.1007/s00607-015-0447-8>. <http://dx.doi.org/10.1007/s00607-015-0447-8>
- Ding S, Xia C, Wang C, Wu D, Zhang Y (2017) Multi-objective optimization based ranking prediction for cloud service recommendation. Decis Support Syst 101:106–114. <https://doi.org/10.1016/j.dss.2017.06.005>
- Doumpos M, Zopounidis C (2002) Multicriteria decision aid classification methods, vol. 73. Springer Science & Business Media
- Ebeling CE (2004) An introduction to reliability and maintainability engineering. Tata McGraw-Hill Education
- EIA: Energy information administration. <http://www.eia.gov/electricity/monthly/> Accessed 20 Feb 2017
- Garg SK, Versteeg S, Buyya R (2013) A framework for ranking of cloud computing services. Futur Gener Comput Syst 29(4):1012–1023. <https://doi.org/10.1016/j.future.2012.06.006>. Special Section: Utility and Cloud Computing
- Ghosh R, Gupta A, Chattopadhyay S, Banerjee A, Dasgupta K (2016) Cocoa: A framework for comparing aggregate client operations in bpo services. In: 2016 IEEE International Conference on Services Computing (SCC). pp 539–546. <https://doi.org/10.1109/SCC.2016.76>
- Gollapudi S (2016) Practical Machine Learning. Packt Publishing Ltd
- Hilber P, Bertling L (2007) Component reliability importance indices for electrical networks. In: Power Engineering Conference, 2007. IPEC 2007. International. IEEE. pp 257–263
- Ishizaka A, Nemery P (2013) Multi-criteria decision analysis: methods and software. Wiley
- Kabir S (2017) An overview of fault tree analysis and its application in model based dependability analysis. Expert Syst Appl 77:114–135. <https://doi.org/10.1016/j.eswa.2017.01.058>

20. Kim DS, Machida F, Trivedi K (2009) Availability modeling and analysis of a virtualized system. In: Dependable Computing, 2009. PRDC '09. 15th IEEE Pacific Rim International Symposium on. pp 365–371. <https://doi.org/10.1109/PRDC.2009.64>
21. Kuo W, Zuo MJ (2003) Optimal reliability modeling: principles and applications. Wiley
22. Lee S, Seo KK (2016) A hybrid multi-criteria decision-making model for a cloud service selection problem using bsc, fuzzy delphi method and fuzzy ahp. *Wirel Pers Commun* 86(1):57–75
23. Liu S, Chan FT, Ran W (2016) Decision making for the selection of cloud vendor: An improved approach under group decision-making with integrated weights and objective/subjective attributes. *Expert Syst Appl* 55:37–47
24. Maciel P (2016) Modeling Availability Impact in Cloud Computing. Springer International Publishing, Cham
25. Maciel P, Trivedi KS, Matias R, Kim DS (2012) Performance and dependability in service computing: Concepts, techniques and research directions. Premier Reference Source. Igi Global
26. Marsan MA, Balbo G, Conte G, Donatelli S, Franceschinis G (1994) Modelling with Generalized Stochastic Petri Nets, 1st edn. Wiley Inc., New York
27. Mell P, Grance T (2009) The nist definition of cloud computing. *Natl Inst Stand Technol* 53(6):50
28. Melo C, Matos R, Dantas J, Maciel P (2017) Capacity-oriented availability model for resources estimation on private cloud infrastructure. In: 2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC). pp 255–260. <https://doi.org/10.1109/PRDC.2017.49>
29. Monczka R, Handfield R, Giunipero L (2008) Purchasing and supply chain management. South-Western Pub
30. Murugesan S, Bojanova I (2016) Cloud Services and Service Providers. Wiley-IEEE Press. <https://doi.org/10.1002/9781118821930.ch2>
31. Pecht M (2009) Product reliability, maintainability, and supportability handbook. CRC Press
32. Rausand M, Hoyland A (2004) System reliability theory: models, statistical methods, and applications, vol. 396. Wiley
33. Rehman Z, Hussain OK, Hussain FK (2012) IaaS cloud selection using mcdm methods. In: e-Business Engineering (ICEBE), 2012 IEEE Ninth International Conference on. pp 246–251. <https://doi.org/10.1109/ICEBE.2012.47>
34. Rehman Z, Hussain OK, Hussain FK (2013) Multi-criteria IaaS service selection based on qos history. In: Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on. pp 1129–1135. <https://doi.org/10.1109/AINA.2013.158>
35. Rehman Zu, Hussain OK, Hussain FK (2013) Parallel cloud service selection and ranking based on qos history. *Int J Parallel Prog* 42(5):820–852. <https://doi.org/10.1007/s10766-013-0276-3>. <http://dx.doi.org/10.1007/s10766-013-0276-3>
36. Sachdeva N, Singh O, Kapur P, Galar D (2016) Multi-criteria intuitionistic fuzzy group decision analysis with topsis method for selecting appropriate cloud solution to manage big data projects. *Int J Syst Assur Eng Manag* 7(3):316–324
37. Sadiku M, Musa S, Momoh O (2014) Cloud computing: Opportunities and challenges. *Potentials*, IEEE 33(1):34–36
38. Sahnner RA, Trivedi K, Puliafito A (2012) Performance and reliability analysis of computer systems: an example-based approach using the SHARPE software package. Springer Publishing Company, Incorporated
39. Sengupta R, Gupta A, Dutta J (2016) Decision Sciences: Theory and Practice. Taylor & Francis Group
40. Silva B, Matos R, Callou G (2015) Mercury: An integrated environment for performance and dependability evaluation of general systems. In: Proceedings of Industrial Track at 45th Dependable Systems and Networks Conference, DSN
41. Sousa E, Lins F, Tavares E, Cunha P, Maciel P (2015) A modeling approach for cloud infrastructure planning considering dependability and cost requirements. *IEEE Trans Syst Man Cybern Syst* 45(4):549–558
42. Trivedi K (2006) Probability and Statistics with Reliability, Queuing, and Computer Science Applications. 2nd Edition. Wiley
43. Trivedi KS, Malhotra M, Fricks RM (1994) Markov reward approach to performability and reliability analysis. In: Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 1994., MASCOTS'94., Proceedings of the Second International Workshop on. IEEE. pp 7–11
44. Upadhyay N (2017) Managing cloud service evaluation and selection. *Procedia Comput Sci* 122:1061–1068. <https://doi.org/10.1016/j.procs.2017.11.474>. 5th International Conference on Information Technology and Quantitative Management, ITQM 2017
45. Wei B, Lin C, Kong X (2011) Dependability modeling and analysis for the virtual data center of cloud computing. In: High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on. IEEE. pp 784–789
46. Weil RL, Maher MW (2005) Handbook of cost management. Wiley

**Submit your manuscript to a SpringerOpen<sup>®</sup> journal and benefit from:**

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

---

Submit your next manuscript at ► [springeropen.com](https://www.springeropen.com)