Proc. 8th IEEE Int. Systems Conference (SysCon 2014),
Ottawa, Canada, March/April 2014, pp. 116-122.

GeoClouds Modcs: A Perfomability Evaluation Tool for Disaster Tolerant IaaS Clouds

Bruno Silva, Paulo Maciel, Jonathan Brilhante
Center for Informatics (CIn)
Federal University of Pernambuco (UFPE)
Recife, Pernambuco
{bs,prmm,jlgapb}@cin.ufpe.br

Armin Zimmermann
System & Software Engineering
Ilmenau University of Technology
Ilmenau, Germany
Armin.Zimmermann@tu-ilmenau.de

***Abstract*—Performance and availability are key aspects to evaluate the quality of cloud computing systems. The assessment of these systems should consider the effects of queuing and failure/recovery behavior of data center subsystems and disaster occurrences. Additionally, penalties may be applied if the defined quality level of SLA contracts is not satisfied. Thus, IaaS providers need to evaluate the performability level of its environment, considering, also, the possibility of disasters. A possible approach to protect cloud systems from natural disasters corresponds to the utilization of redundant data centers located far enough apart. However, the time to back up the VM data increases with the distance. To accomplish these issues, we propose a user-friendly tool, namely GeoClouds Modcs, for evaluating distributed cloud computing systems deployed into multiple data centers considering disaster occurrence. The proposed environment adopts a hybrid heterogeneous modeling approach, which includes Reliability Block Diagrams (RBD), Stochastic Petri Nets (SPN) and Cloud System High-Level models to perform the system evaluation. For specialized users, the tool also provides specific features that enable edit and evaluate the result SPN and RBD models on external evaluation tools (i.e., Mercury and TimeNET). To illustrate the proposed tool's usability, we present a case study that evaluates a cloud computing distributed in different cities considering diverse user loads.**

Keywords*—*performability evaluation tool, IaaS systems, stochastic petri nets

I. Introduction

Cloud computing has driven the new wave of Internet-based applications by providing computing as a service [1]. Nowadays, usual business applications (e.g., spreadsheets, text editors) are provided as cloud computing services, in the sense that they are often accessed using a web browser, and, their respective software/data reside on remote servers. Such paradigm is attractive for a number of reasons: (i) it frees users from installing, configuring and updating the software applications; (ii) it offers advantages in terms of mobility as well as collaboration; and (iii) updates and bug fixes can be deployed in minutes, simultaneously affecting all users around the globe [2].

An important type of cloud service is the Infrastructure-as-a-Service (IaaS), such as Amazon EC2 [3] and IBM Smart Business Cloud [4]. IaaS delivers, on-demand, computing resources in the form of virtual machines (VMs) deployed into the cloud provider's data center, satisfying user needs. User requests are provisioned depending on the data center capacity in terms of physical machines. Considering the user point of view, performability measures are prominent indicators to assess provider's quality-of-service (QoS). These metrics take into account the effects of queuing and failure/recovery behavior of data center subsystems. For prominent IaaS providers, the quality level is regulated by adopting a Service Level Agreement (SLA), which specifies, for instance, the maximum downtime per year. Penalties may be applied if the defined quality level is not satisfied. Thus, to meet SLA requirements, IaaS providers need to evaluate the performability level of its environment, considering, also, the possibility of disasters.

Thus, an analysis of performability is necessary to determine an effective architecture for building a cloud computing infrastructure. It is important to state that the assistance of software tools is very important to obtain performability metrics, since it is not trivial to analyze or simulate complex cloud computing infrastructures. Modeling techniques, with a strong mathematical foundation, such as Stochastic Petri Nets (SPN) [5] and Reliability Block Diagrams (RBD) [6] can be adopted in conjunction to evaluate performability in complex infrastructures.

A disaster recovery plan requires the utilization of different data centers located far enough apart to mitigate the effects of unforeseen disasters (e.g., earthquakes) [7]. If multiple data centers are located in different geographical locations (considering disaster independent places), the availability level of whole system will improve. On the other hand, VM backup time increases due to distance between data centers. Additionally, failures and overloads can lead to system downtime considering cloud computing systems. Consequently, performability evaluation considering VM data backup time and different user loads levels is of utmost importance when considering the analysis of distributed cloud systems.

This work presents a software tool, namely GeoClouds Modcs, to evaluate performability metrics in IaaS systems deployed into geographically distributed data centers as well as taking into account disaster occurrence. The proposed environment contemplates SPN and RBD models to allow performability evaluation. Using the GeoClouds Modcs, IaaS providers can evaluate the system distributed in different data centers and the impact of VM transmission time on performability metrics. The proposed tool supports the methodology presented in our previous works [8] and [9] as well as provide a high level reference model for representing IaaS infrastructures.

The paper is organized as follows. Section II highlights the related works. Section III presents basic concepts related to performability evaluation, describes the cloud computing system and the models utilized by the tool. Then, Section V presents a case study. Finally, Section VI concludes this paper and introduces future works.

II. Related Works

Over the last years, some authors have been devoting efforts to study dependability issues on cloud computing systems. Longo et al. [10] proposed an approach for availability analysis of cloud computing systems based on Petri nets and Markov chains. The authors also developed closed-form equations and demonstrated that their approach can scale for large systems.

In [11], a performability analysis for cloud systems is presented. The authors quantify the effects of variations in workload, failure rate and system capacity on service quality. In [12], the authors investigate the aging effects on the Eucalyptus framework [13], and they also propose a strategy to mitigate such issues during system execution.

[14] describes a system design approach for supporting transparent migration of virtual machines that adopt local storage for their persistent state. The approach is transparent to the migrated VM, and it does not interrupt open network connections during VM migration. In [15], the authors present a case study that quantifies the effect of VM live migrations in the performance of an Internet application. Such study helps data center designers to plan environments in which metrics, such as service availability and responsiveness, are driven by Service Level Agreements.

Dantas et al. [16] presents a study of warm-standby mechanisms in Eucalyptus framework. Their results demonstrate that replacing machines by more reliable counterparts would not produce improvements in system availability, whereas some techniques of fault-tolerance can indeed increase dependability levels.

[17] describes a software testing environment, using cloud computing technology and virtual machines with fault injection facility. The software injects failures in cloud environments in order to evaluate its dependability metrics by using measurements activities. Despite the quality of the work, it is important to stress that the data collecting can be time consuming. Additionally, the tool just can evaluate infrastructures that are already deployed. Hence, it is not possible to perform system evaluation on project design time.

Unlike previous works, this paper proposes a tool for evaluating cloud computing systems deployed into geographically distributed data centers, considering VM migration, disasters occurrence and different user loads. Moreover, performability metrics taking into account user and provider perspectives are adopted by this work.

III. Background and Theory

For a better understanding of the proposed environment, this section presents important concepts for the evaluation of geographically distributed cloud systems.

A. *Performability Evaluation*

Generally, in the evaluation of computing systems, performance and dependability metrics have been adopted considering different perspectives. Performance evaluation takes into account “how well the systems performs” and dependability assessment considers “probabilities of system executing successfully”. However, in degradable systems (in which, the system not necessarily stop working in case of faults) performance and reliability issues must be evaluated simultaneously to access system effectiveness [18].

Performance evaluation refers to a set of techniques and methods which permits assess the temporal behavior of systems. More specifically, the performance evaluation involves the system assessment under a workload [19]. Performance metrics can be evaluated adopting measurements and modeling techniques. The most suitable models to evaluate performance metrics are: Temporal Logics, Networks of Queues and Markov Chain based models (e.g., SPN) [6].

The dependability of a system can be understood as the ability to deliver a set of services that can be justifiably trusted [20]. Indeed, dependability is also related to disciplines, such as security, survivability, reliability and availability. For instance, the reliability of a system at time t' is the probability that the system has not failed from $t = 0$ until $t = t'$, whereas its steady-state availability is the (long-term mean) availability.

State-based models are a suitable choice to model complex interactions between components, such as dynamic redundancy mechanisms and maintenance policies [21]. Markov Chains, Stochastic Petri nets, and Stochastic Process algebras are most widely adopted state-based models for performability evaluation [6]. The reader should refer to [6] and [19] for more details about performance and dependability evaluation.

B. *System Architecture of Reliable Distributed Data Centers*

This section presents an overview of the cloud computing system considered in this work, which contemplates a set of components, distributed over distinct data centers (Figure 1). The system is composed of d data centers, each with two set of machines, namely, hot and warm pools. The hot pool is composed of n physical machines (PM), which are active and running virtual machines (VM). The warm pool consists of m PMs that are active, but without running VMs. Thus, the number of PMs in a data center is $t = m + n$.

Depending on the capacity of each PM, it is possible to run multiple VMs in the same host. In this study, we assume all physical machines are identical, in the sense that they adopt the same services and hardware/software components. PMs may share a common network attached storage (NAS) or a storage area network (SAN) to provide distributed storage and to allow the migration of a virtual machine from one server to another in the same data center [22]. In case of failure, a VM must be instantiated in another physical machine. If there is no available PM, the VM image is migrated to another data center.

Furthermore, a Backup Server (BS) is assumed to provide backup of VM data. This component receives a copy of each VM image during data center operation. Hence, whenever a disaster makes one data center unavailable, BS sends VM copies to an operational data center.

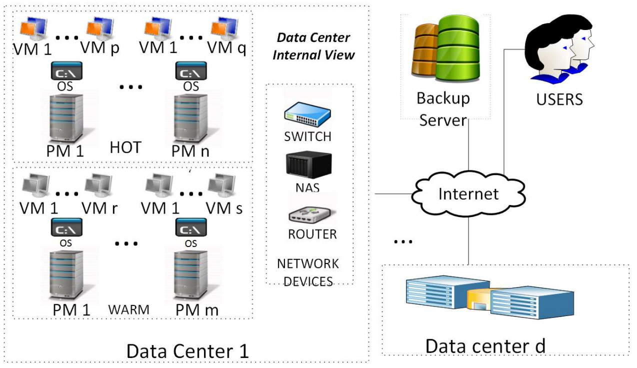


Fig. 1. Distributed Cloud System Example

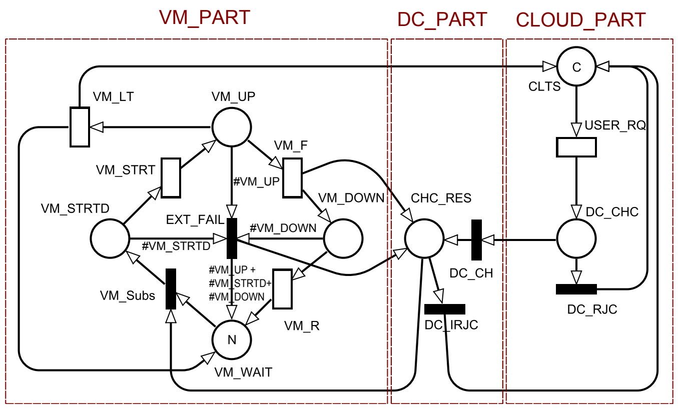


Fig. 3. VM performability SPN model

C. Scalable cloud system models

This section presents the adopted hierarchical modeling to evaluate IaaS cloud systems. The adopted modeling process considers first the evaluation of lower-level submodels, and then the respective results are applied to higher-level models. Dependability results estimated from the RBD models are associated to SPN models. After that, the basic SPN models are configured and combined to create the SPN final model. For further details about the hierarchical modelling approach the reader is referred to [8] and [9].

1) SPN block: generic component: The generic component (Figure 2) is adopted to represent components that have no redundancy and might be in two states, either functioning or failed. In order to compute its availability, mean time to failure (*MTTF*) and mean time to repair (*MTTR*) are the only parameters needed for computing its availability. The respective SPN model of this model is shown in Figure 2.

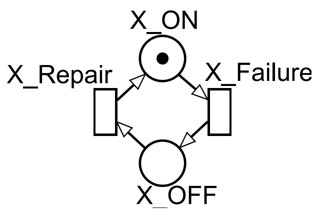


Fig. 2. Generic component SPN model

Places *X_ON* and *X_OFF* are the model component's activity and inactivity states, respectively. Label "X" is instantiated according to the component name, for instance, *DC_UP1* and *DC_DOWN1* (Figure 9). A component is operational only if the number of tokens in place *X_ON* is greater than 0.

2) SPN block: VM performability component: VM performability component represents VM requests on a single physical machine considering failures and repairs on the underlying infrastructure. Whenever a user request is performed, a new VM is started (considering that the infrastructure is operational) and becomes available for some time. This component interacts with three generic components: (i) one representing the occurrence of disasters (*DC*); (ii) the network infrastructure (*NAS_NET*); and (iii) the physical machine (*OSPM*).

If the external infrastructure or the VM fail during the virtual machine execution, the VM should be migrated to another physical machine. In case of no available physical machine in the system, the task is rejected and a new request must be performed. Figure 3 presents the VM performability model, which is composed of three main parts: (i) *VM_PART* represents the VM behavior running on a single machine; (ii) *DC_PART* which express the incoming requests to data center; and (iii) *CLOUD_PART* that models the requests generation.

3) SPN block: transmission component: A VM should migrate to another data center whenever the current data center is full or the underlying infrastructure is failed. Moreover, Backup Server is responsible for transmitting the VM image in case of disaster or network error. Transmission component represents the data transfer from a data center to another and the data transfer from Backup Server to a data center.

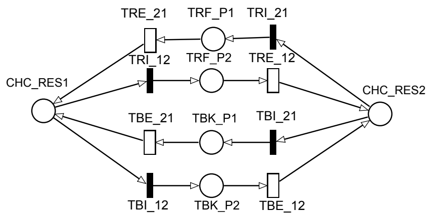
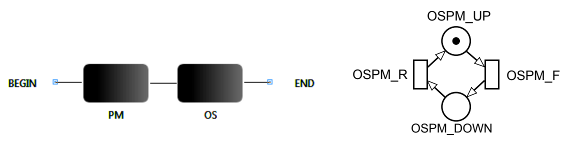


Fig. 4. Transmission component SPN model

4) RBD modeling approach: The adopted modeling process considers first the evaluation of lower-level submodels, then the respective results are applied to higher-level models. For instance, Figure 5 depicts a RBD model, such that the operating system (OS) and the physical machine (PM) are in series arrangement.

MTTR and *MTTF* results estimated from the RBD [21] are associated to transitions *OSPM_R* and *OSPM_F*, respectively, of the SPN model depicted in Figure 5(b). The modeling approach contemplates RBD models for representing the physical machine (*OS_PM*) as well as the data center network infrastructure (*NAS_NET*), such that the respective MTTFs and MTTRs are estimated and utilized in generic component models (Section III-C1).

5) Proposed Metrics: To perform system evaluation we consider provider and client oriented metrics. The provider's oriented metric is the machine utilization in terms of maximum



(a) RBD model (b) SPN model

Fig. 5. RBD model and the respective SPN model for operating system and physical machine.

number of VMs (U) and the user's oriented metric is the probability of a task be completed without error (P).

The following operators are adopted for assessing the metrics: $P\{exp\}$ estimates the probability of the inner expression (exp); $\#p$ denotes the number of tokens in place p and $E\{\#p\}$ estimates its expected number. U is calculated as follows:

$$U = \frac{\sum_{j=1}^{M} E\{\#VM\_UPj\}}{M.N} \quad (1)$$

.

Where M corresponds to the number of PMs and N is the maximum number of VMs per physical machine. P corresponds to:

$$P = \frac{(\sum_{j=1}^{Pm} E\{\#VM\_UPj\}).(1/T)}{P\{\#CLTS > 0\}.(1/R)} \quad (2)$$

.

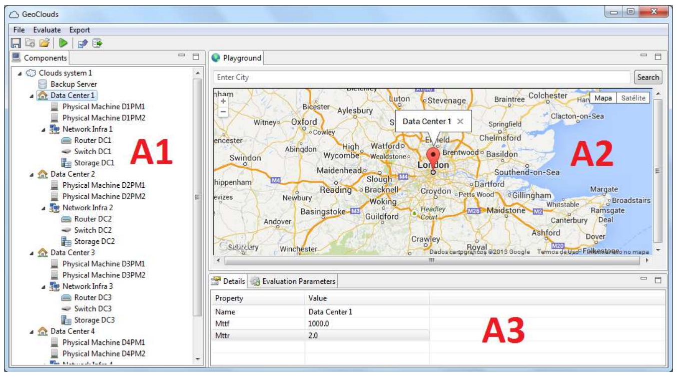
In which T and R correspond to the times associated with transitions *VM_LT* and *USER_REQ* (Figure 3). This expression divides the throughput of completed requests by the incoming requests throughput to compute the percentage of completed requests.

IV. Environment Overview

As previously stated, GeoClouds Modcs is an integrated environment, which performs dependability and performance evaluation for geographically distributed IaaS infrastructures. The environment adopts RBD and SPN for performability assessment. It is important to state that GeoClouds Modcs relies on Mercury tool [23] as a standard engine to perform the evaluation process. However, other prominent tools, such as TimeNET [24], can be adopted as evaluation engines. The tool is written in Java and utilizes Google Maps API to select the data center positions.

Figure 6 presents the main screen of GeoClouds Modcs. The tool presents three main areas: A1 - for representing the IaaS infrastructures and subcomponents, A2 - in which the user can select the data center position and A3 - where the architecture and evaluation parameters are detailed.

For a better understanding of the evaluation process, Figure 7 depicts the data flow of the tool. Firstly, the user configure the High Level IaaS model providing the parameters and the evaluation details. After the user's evaluation request, the tool creates and evaluates an internal RBD model. The RBD results are utilized in the SPN model to configure the servers and network infrastructures (See Section III-C4). Finally, the SPN model is evaluated and the performability results are provided.



Property	Value
Name	Data Center 1
Mttf	1000.0
Mttr	2.0

Fig. 6. Tool Overview

It is important to stress that the geographical information is utilized to estimates the mean time to transmit the VM data (*MTT*) from one geographical place to another. If the user already knows the actual transfer rate (Bytes/s), it can be provided directly and no geographical information is needed.

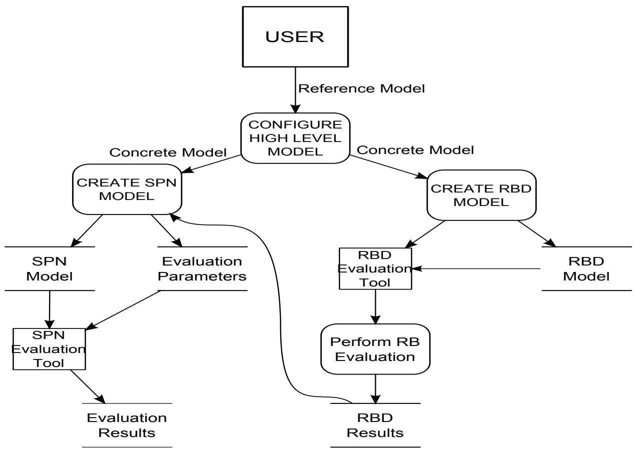


Fig. 7. Data Flow Model

A. Mean Time to VM Transmit (MTT) estimation

To estimate the *MTT* value, we consider the approach presented in [25] that provides an equation to assess the network throughput based on the distance between the communication nodes. The VM transmission rate is obtained as follows:

$$Rate < (MSS/RTT) \times (1/\sqrt{p}) \quad (3)$$

where *Rate* (kbps) is the TCP transfer rate, *MSS* (bytes) is the maximum segment size per package, *RTT* (ms) is the round trip time, and *p* is the packet loss ratio. To estimate the RTT we following equation is adopted:

$$RTT = \frac{Dist}{\alpha \times 100} \quad (4)$$

in which *Dist* means the distance in kilometers. The equation associates a constant α with network directness, in which values close to one mean the path between the hosts follows a direct path. Values much smaller than one mean the path is very indirect. The values of α, MSS and p are user configurable. The distance (*Dist*) is taken from Google Maps API. Further GeoClouds Modcs versions will provide a component to assist the users to estimate α values.

B. High Level IaaS Model

This section presents the proposed high level model. It is important to state that the user provides the reference model using the graphical interface and he/she does not need to know details about mathematical representation. A geographically distributed IaaS system corresponds to the tuple $G = (F_{lt}, T_{di}, T_{re}, T_{tf})$ in which:

- F_{lt} is a finite set of facilities, including data centers and backup servers, so that $F_{lt} = D \cup B$. D is a finite set of data centers and B represents the unitary set ($|B| = 1$) of backup servers.
- T_{di} (so that, $T_{di} : F_{lt} \to \mathbb{R}$) represents the mean time to disaster function.
- T_{re} (where, $T_{re} : F_{lt} \to \mathbb{R}$) represents the mean time to recovery function.
- T_{tf} (so that, $T_{tf} : F_{lt} \times F_{lt} \to \mathbb{R}$) represents the mean rate to transfer a VM from a facility to another.

A data center $dc \in D$ corresponds to ordered pair (P_d, C_d), where P_d represents a represents a physical machine finite set. C_d represents the finite set of basic components of network infrastructure.

A physical machine $p \in P_d$ corresponds to the tuple (V_p, S_p, os, hw, m) where:

- V_p represents a virtual machine finite set related to the physical machine at cloud system start up.
- S_p (so that, $S_p : V_p \to \mathbb{R}$) denotes the virtual machine mean set up time.
- $os \in O_p$ corresponds to the physical machines's software component.
- $hw \in H_p$ represents the hardware of the physical machine.
- $m \in \mathbb{N}$ denotes the maximum of virtual machines that the physical machine can execute.

O_p and H_p are finite sets of the software and hardware components related to physical machines. C ($C = C_d \cup O_p \cup H_p \cup V_p$) corresponds to a finite set of all data center's basic components. T_{fr} represents the mean time to failure associated with a component $c \in C$ so that $T_{fr} : C \to \mathbb{R}$. T_{rp} represents the mean time to failure associated with a component $c \in C$ so that $T_{rp} : C \to \mathbb{R}$.

V. Case Study

This section presents a case study to illustrate the importance of software tools for helping cloud computing designers to estimate performability metrics. Figure 8 presents a cloud architecture located in Brazil, which is composed of a data center located in Recife (Data Center 1), other in Rio de Janeiro (Data Center 2) and a Backup server in São Paulo. Each data center consists of two physical machines and each machine is capable to run up two virtual machines.

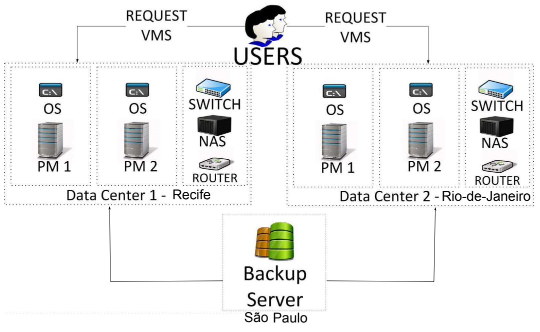


Fig. 8. Cloud System Architecture

Table I presents the dependability parameters associated with the devices, which were taken from [26], [27], [28], [29]. We assume the package loss ratio 1%, MSS 1460 bytes, α as 0.45 and the size of VMs as 4GB. We considered the mean time between disaster to be 100 years and the data center to take 30 days to be repaired. Moreover, a VM takes five minutes to set up and the mean time between requests is half an hour. The mean time for using a VM is 720 hours.

TABLE I. Dependability parameters for components of Figure 9.

Component	MTTF(h)	MTTR(h)
Operating System (OS)	4000	1
Hardware of Physical Machine (PM)	1000	12
Switch	430000	4
Router	14077473	4
NAS	20000000	2
VM	2880	0.5
Backup Server	50000	0.5

In order to perform the evaluation, the user should provide the parameters and execute the evaluation. The assessment process can be conducted in a transparent way directly on the tool or adopting an external tool such as Mercury [30] or TimeNET [24].

The generated SPN model is presented in Figure 9, which is composed of four VM performability models, one transmission component as well several generic components. *OSPM_1* and *OSPM_2* represent the physical machines of Data Center 1, and *OSPM_3* as well as *OSPM_4* are the models related to PMs of Data Center 2. *DISASTER1* and *DISASTER2* models disasters in Data Centers 1 and 2, respectively. *NAS_NET_1* and *NAS_NET_2* corresponds to network devices of Data Center 1 and 2.

Figure 10 shows VM utilization and the rejection probability for different number of clients. It is assumed that each client can request a single VM. The results show that the utilization increases with the number of clients. As the task is rejected if the infrastructure is broken or the servers are full, it is possible to observe that the rejection probability increase due to increase of system load. Therefore, for this particular system and user loads, we can conclude that the rejection probability is highly impacted by server utilization.

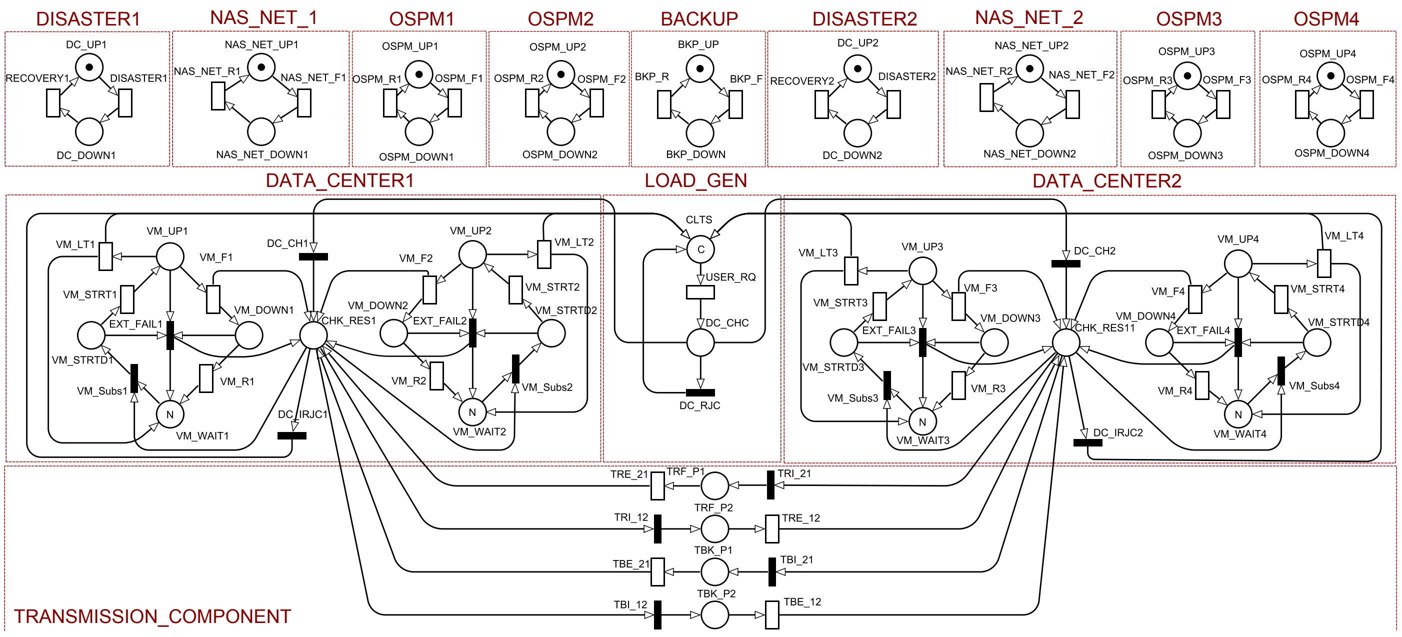


Fig. 9. Case study output model

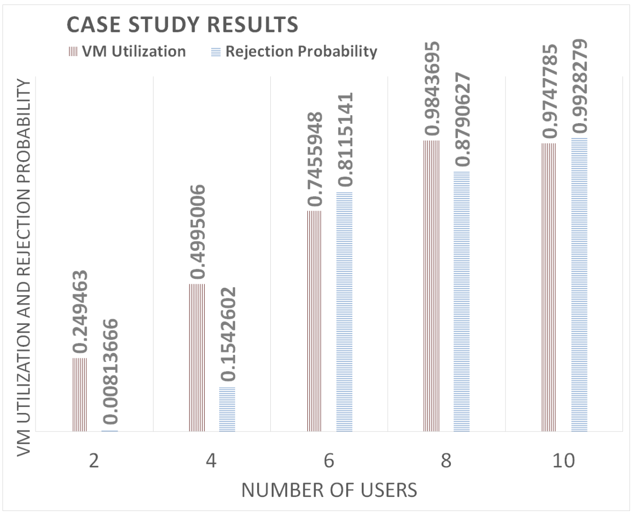


Fig. 10. Case Study Results

VI. Conclusion

This work presented an evaluation tool (GeoClouds Modcs) for performability evaluation of cloud computing systems deployed into geographically distributed data centers taking into account disaster occurrence. The proposed environment allows the impact assessment of disaster occurrence, VM transmission time and different client loads on system performability. Additionally, a case study is provided considering data centers located in different places around the world.

The results demonstrated the client load on performability metrics taking into account disaster occurrence. User and provider oriented metrics are adopted to show the trade-off between dependability and user loads in the case study. As future research, we intend to evaluate costs and environmental impacts considering the proposed infrastructure. Additionally, we intend to study the effects of distance between data centers to meet recovery plan constraints, such as Recovery Point Objective (RPO) and Recovery Time Objective(RTO).

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010. [Online]. Available: http://doi.acm.org/10.1145/1721654.1721672

[2] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, pp. 7–18, 2010. [Online]. Available: http://dx.doi.org/10.1007/s13174-010-0007-6

[3] Amazon ec2. [Online]. Available: http://aws.amazon.com/ec2

[4] IBM smart business cloud. [Online]. Available: http://www-935.ibm.com/services/us/igs/cloud-development/

[5] A. Zimmermann, *Stochastic Discrete Event Systems: Modeling, Evaluation, Applications*, ser. SpringerLink: Springer Books. Springer, 2008.

[6] P. Maciel, K. S. Trivedi, R. Matias, and D. S. Kim, *Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions*, ser. Premier Reference Source. Igi Global, 2011, ch. Dependability Modeling.

[7] *Hyper-V Live Migration over Distance*. [Online]. Available: http://goo.gl/GzlkNk

[8] B. Silva, P. R. M. Maciel, E. Tavares, and A. Zimmermann, "Dependability models for designing disaster tolerant cloud computing systems," in *The Third International Workshop on Dependability of Clouds, Data Centers and Virtual Machine Technology (DCDV)*, 2013.

[9] B. Silva, P. R. M. Maciel, and A. Zimmermann, "Performability models for designing disaster tolerant infrastructure-as-a-service cloud computing systems," in *The 8th International Conference for Internet Technology and Secured Transactions (ICITST)*, 2013.

[10] F. Longo, R. Ghosh, V. Naik, and K. Trivedi, "A scalable availability model for infrastructure-as-a-service cloud," in *Dependable Systems Networks (DSN), 2011 IEEE/IFIP 41st International Conference on*, june 2011, pp. 335 –346.

[11] R. Ghosh, K. S. Trivedi, V. K. Naik, and D. S. Kim, "End-to-end performability analysis for infrastructure-as-a-service cloud: An interacting stochastic models approach," in *Proceedings of the 2010 IEEE 16th Pacific Rim International Symposium on Dependable Computing*, ser. PRDC '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 125–132. [Online]. Available: http://dx.doi.org/10.1109/PRDC.2010.30

[12] J. Araujo, R. Matos, P. Maciel, R. Matias, and I. Beicker, "Experimental evaluation of software aging effects on the Eucalyptus cloud computing infrastructure," in *Proceedings of the Middleware 2011 Industry Track Workshop*, ser. Middleware '11. New York, NY, USA: ACM, 2011, pp. 4:1–4:7. [Online]. Available: http://doi.acm.org/10.1145/2090181.2090185

[13] "Open source private and hybrid clouds from Eucalyptus," http://www.eucalyptus.com.

[14] R. Bradford, E. Kotsovinos, A. Feldmann, and H. Schiöberg, "Live wide-area migration of virtual machines including local persistent state," in *Proceedings of the 3rd international conference on Virtual execution environments*, ser. VEE '07. New York, NY, USA: ACM, 2007, pp. 169–179. [Online]. Available: http://doi.acm.org/10.1145/1254810.1254834

[15] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of virtual machine live migration in clouds: A performance evaluation," in *Proceedings of the 1st International Conference on Cloud Computing*, ser. CloudCom '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 254–265.

[16] J. Dantas, R. Matos, J. Araujo, and P. Maciel, "An availability model for eucalyptus platform: An analysis of warm-standy replication mechanism," in *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*, oct. 2012, pp. 1664 –1669.

[17] T. Banzai, H. Koizumi, R. Kanbayashi, T. Imada, T. Hanawa, and M. Sato, "D-cloud: Design of a software testing environment for reliable distributed systems using cloud computing technology," in *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, ser. CCGRID '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 631–636. [Online]. Available: http://dx.doi.org/10.1109/CCGRID.2010.72

[18] J. Meyer, *On Evaluating the Performability of Degradable Computing Systems*, ser. R, 1978.

[19] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*, ser. Wiley Professional Computing. Wiley, 1991.

[20] A. Avizienis, J. Laprie, and B. Randell, "Fundamental Concepts of Dependability," *Technical Report Series-University of Newcastle upon Tyne Computing Science*, 2001.

[21] C. Ebeling, *An Introduction to Reliability and Maintainability Engineering*. Waveland Press, 1997.

[22] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2*, ser. NSDI'05. Berkeley, CA, USA: USENIX Association, 2005, pp. 273–286. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251203.1251223

[23] "Mercury environment." [Online]. Available: http://www.modcs.org

[24] R. German, C. Kelling, A. Zimmermann, and G. Hommel, "TimeNET: a toolkit for evaluating non-markovian stochastic petri nets," *Performance Evaluation*, vol. 24, no. 1-2, pp. 69 – 87.

[25] W. M. Les Cottrell and C. Logg, "Tutorial on internet monitoring and pinger at SLAC," Tech. Rep., 1996. [Online]. Available: http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html

[26] D. S. Kim, F. Machida, and K. S. Trivedi, "Availability modeling and analysis of a virtualized system," in *Proceedings of the 2009 15th IEEE Pacific Rim International Symposium on Dependable Computing*, ser. PRDC '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 365–371.

[27] (2012, Oct.) Cisco systems: Switch dependability parameters. [Online]. Available: http://tinyurl.com/cr9nssu

[28] (2012, Oct.) Cisco systems: Router dependability parameters. [Online]. Available: http://tinyurl.com/d7kcnqo

[29] (2012, Oct.) Service level agreement - megapath business access and value added services. [Online]. Available: http://tinyurl.com/cwdeebt

[30] B. Silva, G. Callou, E. Tavares, P. Maciel, J. Figueiredo, E. Sousa, C. Araujo, F. Magnani, and F. Neves, "Astro: An integrated environment for dependability and sustainability evaluation," *Sustainable Computing: Informatics and Systems*, 2012.