JOURNAL OF INFORMATION SYSTEMS ENGINEERING & MANAGEMENT

Journal of Information Systems Engineering & Management, 2018, 3(3), 25
ISSN: 2468-4376

lectito

# Redundancy Mechanisms Applied in Video Streaming Service on Eucalyptus Cloud Computing Infrastructures

Rosangela Melo [1, 2*], Vicente de Paulo F. Marques Sobrinho [3], Ivanildo José de Melo Filho [1, 2], Fábio Feliciano [1, 4], Paulo Romero Martins Maciel [1]

[1] *Informatics Center, Federal University of Pernambuco,* Recife, BRAZIL
[2] *Federal Institute of Education, Science, and Technology of Pernambuco (IFPE),* Paulista Campus, BRAZIL
[3] *Federal Institute of Education, Science, and Technology of Espirito Santo (IFS),* Vitória Campus, BRAZIL
[4] *Federal Institute of Education, Science, and Technology of Pernambuco (IFPE),* Belo Jardim Campus, BRAZIL

*__Corresponding Author:__ rosangela.melo@paulista.ifpe.edu.br

**Citation:** Melo, R., Sobrinho, V. P. F. M., Filho, I. J. M., Feliciano, F. and Maciel, P. R. M. (2018). Redundancy Mechanisms Applied in Video Streaming Service on Eucalyptus Cloud Computing Infrastructures. *Journal of Information Systems Engineering & Management*, 3(3), 25. https://doi.org/10.20897/jisem/2663

**Published:** July 16, 2018

## ABSTRACT

Architectures and services that are provided by cloud computing system must have high availability, scalability, security and furthermore be fault tolerant. Plan these environments is not an easy task, it is necessary to ensure that undesirable situations or errors do not occur or can be minimized. The use of Sensitivity Analysis combined with the use of modeling hierarchy and the study of models for representing redundancy mechanisms arise in order to investigate the possible changes that these systems suffer and identify their shortcomings and propose improvements solutions for the planning of these systems. In this work the redundancy mechanisms is a solution to improve the performance of environments in cloud computing.

**Keywords:** availability, redundancy mechanisms, sensitivity analysis, video streaming service

## INTRODUCTION

Architectures and services that are provided by cloud computing system must have high availability, scalability, security and furthermore be fault tolerant. Plan these environments is not an easy task, it is necessary to ensure that undesirable situations or errors do not occur or can be minimized. Redundancy is apply in large systems requiring high reliability such as infraestruture and services hosted in cloud computing environment. Therefore sensitivity analysis strategies can be applied to these environment in order that these characteristics may be guaranteed (Pannell, 1997).

In this paper, we propose an availability model of the VoD service based on Eucalyptus cloud environment, with the objective of maintaining the metric availability of these infrastructures. In order to do this, we use the sensitivity analysis strategies associated to the use of hierarchical modeling and the study of models to represent the redundancy mechanisms of the cloud computing environments to identify the critical points that require improvement in the availability of the system.

The remainder of the paper is organized as follows: Section II introduces related works on system availability and sensitivity analysis, Section III introduces basic concepts of cloud computing technologies, video streaming, dependability models and sensitivity analysis. Section IV presents the architecture of the system analyzed in this paper. Section V presents the availability model designed for architecture. Section VI conducts a case study about

architecture analysis and presents sensitivity analysis. Finally, Section VII shows the conclusions of the study and suggests possible future work.

## RELATED WORKS

Over the last years, some works have been intended to evaluate dependability of cloud infrastructures. They have employed hierarchical modeling to represent cloud computing architectures, it is possible to compare the different solutions and estimations of dependability measures (Dantas et al., 2012; Chuob et al., 2011). The modeling approach also allows the evaluation of cloud infrastructures with different redundancy mechanisms concerning dependability. Moreover, some works have also employed sensitivity analysis to identify the critical system components, and thereby propose infrastructure improvements (Matos et al., 2012).

Yefremov (2005) shows a work dedicated to reliability analysis of parallel redundant configurations using Petri nets simulation. Generalized model for hot, warm and cold standby redundancy with or without repair is presented. Khazaei et al. (2012) integrated an availability model in overall analytical submodels of cloud system. Each sub-model captures a specific aspect of cloud centers. The key performance metrics such as task blocking probability and total delay incurred on user tasks are obtained.

Dantas et al. (2012) investigated the benefits of a warmstandby replication mechanism in a Eucalyptus cloud computing environment. A hierarchical modeling approach was used to represent a redundant architecture and compare its availability to that of a non-redundant architecture.

On the other hand, Matos et al. (2012) investigated on the availability of data networks, including redundancy mechanisms. Several scenarios are evaluated through analytic-numeric solution of Markov chains. Furthermore, the impact of different component parameters on the overall system availability was evaluated, by means of differential sensitivity analysis. In this paper, the authors proposed availability models applied to a cloud environment for VoD streaming service with redundancy mechanisms and sensitivity analysis like strategy of planning for cloud environment applied a Video streaming service system.

## FUNDAMENTAL CONCEPTS

This section presents the concepts which provide a background for this paper, including: cloud computing technologies, dependability modeling, mechanisms of redundancy and sensitivity analysis techniques.

### Cloud Computing

A cloud computing system is comprised of a bundle of resources, such as hardware, software, development platforms and services, readily usable and accessible through the Internet (Armbrust et al., 2010). Services can be provisioned on different levels by cloud computing providers, including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) (See **Figure 1**). In the IaaS level, provider has the ability to offer a processing and storage infrastructure transparently. The user at this level do not have control of the physical infrastructure but through virtualization mechanisms, has control over the virtual machines, storage, installed applications, and possibly limited control of network resources (Armbrust et al., 2010).

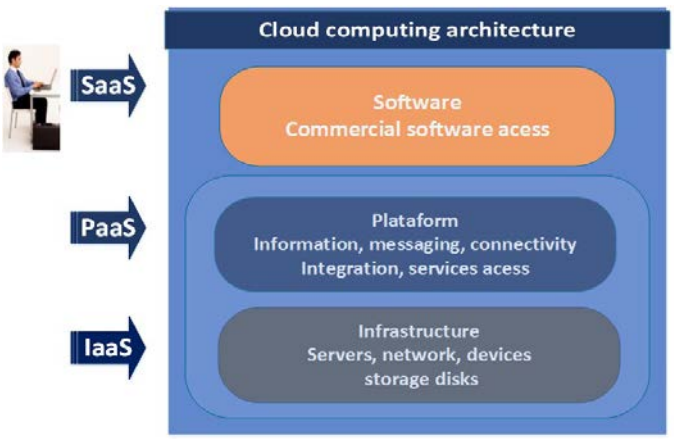


**Figure 1.** Cloud computing architecture (Eucalyptus cloud computing platform, 2010)

On the other hand, the PaaS level provides the services so that applications can be developed, implemented, tested and brought to the cloud environment by service providers. On this level of development, end users do not

 

have access. The access is only for user more experienced, developers of solutions for cloud computing. We found these interfaces Web 2.0 environments, distributed resources and programming languages (Armbrust et al., 2010). In the last, the SaaS level the applications are offered as services by providers and the applications are accessed by customers through of browser (Armbrust et al., 2010).

**Eucalyptus Platform**

Eucalyptus is a software architecture based on Linux that leverage the implementation of private and hybrid clouds. This means that users can utilize their own collections of resources (hardware, storage and network) through a selfservice interface according to their needs. The Eucalyptus software framework is modular (Eucalyptus cloud computing platform, 2010), and consists of five high level components, each with its own Web Service: Cloud Controller (CLC), Cluster Controller (CC), Node Controller (NC), Storage Controller (SC), e Walrus (Eucalyptus cloud computing platform, 2010) (see **Figure 2**).

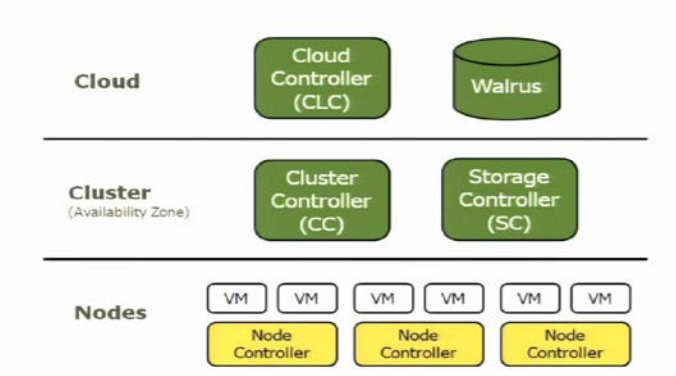


**Figure 2.** Eucalyptus platform architecture (Eucalyptus cloud computing platform, 2010)

The CLC is the frontend of the cloud infrastructure. It employs web service interfaces to receive client tool requests on one side and interact with the remaining Eucalyptus components on the other side (Dantas et al., 2012).

**Models for Dependability Analysis**

Dependability is related to disciplines such as fault tolerance and reliability. The concept of dependable computing first appeared in 1820s when Charles Babbage undertook the enterprise to conceive and construct a mechanical calculating engine to eliminate the risk of human errors (Laprie, 1995; Schaffer, 1994; Maciel et al., 2011). Availability can be expressed as the ratio of the expected system uptime to the expected system up and downtime: A = E[Uptime]/(E[Uptime]+E[Downtime]) It may also be represented by: A = MTTF/(MTTF +MTTR) where MTTF and MTTR is the mean time to failure and to recovery, respectively.

There are several types of models that can be used for analytical evaluation of dependability. Reliability Block Diagrams (RBD), Fault Trees, Stochastic Petri Nets (SPN) and Continuous Time Markov Chains (CTMC) have been used to model fault-tolerant systems and evaluate various dependability measures (Maciel et al., 2011).

**Mechanisms of Redundancy**

The redundancy mechanisms can be classified as activeactive and active-standby. The redundancy mechanisms activeactive type are used when the primary and secondary components share the system workload. When any of these components fails, the other component will be the responsibility 'requests for the service of the system users. These redundancy mechanisms can be classified as N + K, where K secondary components identical to N primary components are required for sharing the workload system. In the N + 1 configuration, a secondary component identical to the N primary components is required to share the system load (Bauer et al., 2011).

On the other hand, the active-standby redundancy type mechanisms are employed when the primary components meet the requests of system users and secondary components are on hold. When the primary component fails, the secondaries will be Responsible for serving the requests of system users. The active-standby redundancy mechanisms can be categorized the hot standby, cold standby and warm standby (Bauer et al., 2011).

In the hot standby redundancy mechanism, redundant modules that are in standby, they are working in sync with the operating module without that your computing is considered in the system, and if the occurrence of a failure event is detected, it is ready to become immediately operation. Active–active redundancy means that traffic is evenly shared by two operational units, but full traffic load can be served with acceptable quality by a single unit (Bauer et al., 2011). In the redundancy mechanism cold standby, redundant modules are switched off only when a failure event occurs is that they are activated after a time interval. In the warm standby redundancy mechanism,

 

redundant modules are in standby are working in sync with the operating module without that your computing is considered in the system, and if a failure event is detected, it is ready to become operational after an interval of time (Bauer et al., 2011).

### Sensitivity Analysis

Sensitivity analysis can assist in the identification of the important components for the dependability and performance, captured in an analytical model. The techniques that can be applied to analyze the sensitivity are: the experimental factorial design, correlation analysis, regression analysis, perturbation analysis (PA), Discrete Averaged Sensitivity Index (DASI) and the parametric differential analysis (PDS), also known as or direct method (Hamby, 1994; Matos et al., 2012).

PDS and DASI strategies of sensitivity analysis was chosen for this work because it can be efficiently performed in analytical models usually employed in availability and performance studies. DASI strategy is very similar a PDS strategy, however it has a significant difference which consists in the possibility of varying the input parameters for the sensitivity analysis.

1) Parametric Differential Sensitivity Analysis - PDS: The differential analysis is accomplished by calculating the partial derivatives for the measures of interest the respective parameters of interest. For example, considering a metric Y which depends on a parameter (λ), the sensitivity of Y with respect to λ is computed with Equation 1, or 2 when adopting scaled sensitivity (Matos et al., 2014).

$$S_\lambda(Y) = \frac{\partial Y}{\partial \lambda} \tag{1}$$

$$S^*_\lambda(Y) = \frac{\partial Y}{\partial \lambda}\left(\frac{\lambda}{Y}\right) \tag{2}$$

2) Discrete Averaged Sensitivity Index (DASI): Discrete Averaged Sensitivity Index (DASI) is the average value of the partial derivatives computed throughout a discrete range of values for the parameter of interest. DASI proposes to reduce dependency associated to a single parameter in the partial derivative method enabling to build unified ranking sensitivity accurately. This sensitivity index can be obtained through Equation 3:

$$DASI_\theta(Y) = \sum_{i=1}^{n} \frac{\partial Y}{\partial \theta} | \theta = \theta i \tag{3}$$

## SYSTEM ARCHITECTURE

### Basic Architecture

The system architecture is based on Eucalyptus cloud computing platform. A broad view of the components of the VoD service architecture is seen in **Figure 3** (Melo et al., 2014; Melo et al., 2017). This architecture consists of a machine for the Frontend and another machine for the Node. The VoD architecture is divided into two sides: client and the server. The physical structure is composed of three machines. One machine is used for frontend and two machines for the nodes. The client connects to the video streaming server through the Internet. A storage volume is allocated in the frontend for storing the collection of videos. A Virtual Machine (VM), running the Apache and VLC applications, is instantiated in the nodes. VLC provides the video streaming features, whereas Apache is responsible for hosting the service on a dedicated Web page. The user issues a request for displaying a video hosted on a specific web page. VLC, in turn, grabs the requested video from the remote storage volume and transmits the stream to the user.

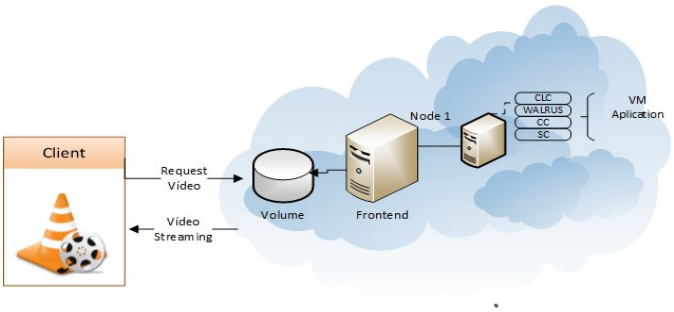


**Figure 3.** Baseline Architecture for VoD service

### Cold Standby Architecture

This architecture very similar the **Figure 3** the difference being, however, consists of a a machine for the Frontend and two machines for the Nodes (1 and 2), operating in cold standby mode. The secondary Node (Node 2) is in the cold standby (turned off:) mode and to turn it on it will take some time that we call 'δ' for the machine to start up and take over the workload. In addition, the machine, which is the Frontend, is responsible for controlling the Nodes.

## AVAILABILITY MODELS

This section discuss the availability models for the architecture baseline of **Figures 3** and architecture models with mechanisms redundancy cold standby of **Figure 4** (Melo et al., 2017), employed to represent the architectures evaluated by this research, from which the availability values were calculated. Such values were obtained by the hierarchical combinatorial method, which combines the system state representation of Markov chains and RBDs (Kim et al., 2009), and is the method commonly employed to evaluate complex IT systems. The systems were modeled and evaluated with the Mercury (Silva et al., 2012) tool, which were specifically designed for the analysis of such models.

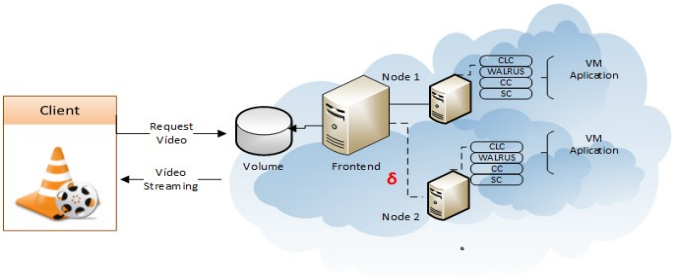


**Figure 4.** Cold Standby Achitecture

### Model for Baseline Architecture

The architecture subsystems in **Figure 3** are represented with RBDs.The basic architecture, as depicted in the top-level RBD of **Figure 5**, is divided into four parts; Frontend, Node, Volume, and Service. Further details about this architecture are available in (Melo et al., 2014).

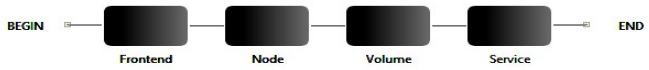


**Figure 5.** Baseline architecture RBD model

### Model for Cold Standby Architecture

Initially, we will use a high level RBD model (see **Figure 6**) to represent the availability of the Video Streaming service hosted on a cloud infrastructure (see **Figure 4**). From this perspective, cold standby redundancy is implemented. The RBD block for the SSMARKOV module shown in **Figure 6** depicts an architecture that has one machine for the Frontend and two machines for the Nodes, operating in cold standby mode. This architecture is represented by the RBD block for the SSMARKOV module. The RBD block for the Volume module, in series with SSMARKOV, represents the video storage device.


**Figure 6.** RBD model for video streaming system

The availability of the system represented by the model of **Figure 6** is obtained through Equation 4.

$$A_s = A_{SSMARKOV} \times A_{VOLUME} \quad (4)$$

Due to some peculiarities that only the redundancy model has, the RBD model can not represent these details, so it is necessary to refine the RBD block to the SSMARKOV module for a Markov chain. The particularities associated with the cold standby model are related to the activation time of the machine that is offline, that is, turned off waiting for a failure of the machine that is running, so that it can enter in operation. In addition, the need to represent the status of the components, such as the UP condition represents the component in operation, condition D (off) represents the component off or fault of the Frontend components, main and secondary Node, offline represents secondary Node offline. Thus, the RBD SSMARKOV block was represented by the Markov chain in **Figure 7**. This model consists of 8 places: UOU, UOD, DOU, DOD, DUU, DUD, DDU and DDD. The states in gray are those that correspond to the system in the operational module while the transparent states correspond to the system in non-operational mode.

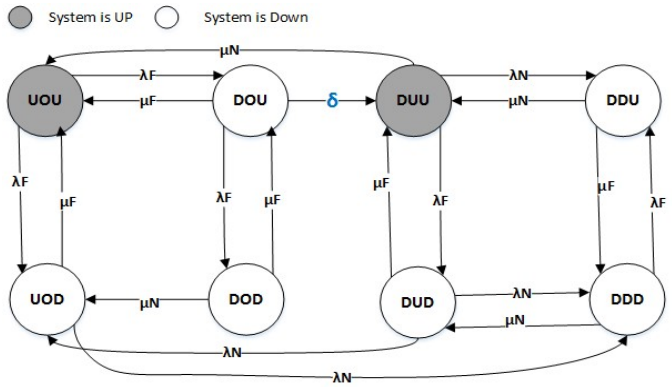


**Figure 7.** Cold Standby CTMC Model

The notation for states is based on the current condition of each machine. The three letters represent Node principal in the situation UP or Off, Node in the condition Offline, UP or Off is the initial state of Frontend in the Off state or UP respectively. In the Eucalyptus platform the machine corresponding to the Frontend holds the control of the machines where Nodes are allocated, at the moment that machine fails we do not have control of the system taking it to the state of unavailability of the service.

In the UOU state the system is with the main Node and Frontend in the Up condition representing the working system, the secondary Node offline. From this state, it is possible to reach the DOU and UOD states. Reaching the UOD state we find the main Node, the secondary Node texting, and the Frontend in the fault state through the $\lambda_f$, and it is possible to perform its repair by the rate mf. When the UOU state reaches the DOU state, it is seen that the main Node is down using the fault rate $\lambda_N$, and can be repaired by the repair $\mu_N$, the secondary node is off and Frontend is Up. Through the UOD state we can reach the DDD state from the failure of the main Node.

From the DOU state, three paths are possible UOU, DOD, DUU. Upon reaching the UOU state the repair of the main Node is achieved through the repair rate $mu_N$. Arriving in the DOD state, the fault occurs in the Frontend, from this state I can reach the UOD state through the recovery of the main Node. From the DOU state to the DUU state, the secondary Node is activated. In this state the secondary Node is in cold standby mode and to switch on it will take a certain amount of time to call $\delta$ for the machine to start up and take over of work.

In the DUU state the system is operational with the Node Up and the Node main down and the Frontend in the operational state, from that state it is possible reach the DUD and DDU states and also UOU. Going to the DDU state fails the secondary Node, in which case we will have both Nodes unavailable and Frontend state is operational, but the service will not be operational. Reaching the main DUD state state is down and the secondary Node UP, but down leading the service to unavailability. Upon reaching the DDD state we have the three components down, the two Nodes (main and redundant), and the Frontend. The Equation of availability for this model of **Figure 7** is obtained through Equation 5:

$$A = P(UOU) + P(DUU) \tag{5}$$


**Figure 8.** RBD model for Frontend

In Frontend subsystem is composed is represented by a pure series RBD, as shown in **Figure 8**. This subsystem consists of Hardware, Operating System, and the following Eucalyptus components: CLC (Cloud Controller), CC (Cluster Controller), SC (Storage Controller) and Walrus.

**Figure 9** represents the RBD model of the Node component. In addition to the Hardware (HW) and Operating System (OS) also present in Frontend, each node requires a hypervisor (KVM) and the Node Controller (NC) of Eucalyptus jamilson12, VM (Virtual Machine) and the VLC and Apache applications, these last three components represent the service.

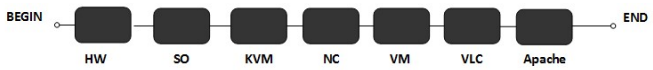


**Figure 9.** RBD model for the Node

This model was used to calculate the mean time of failure (MTTF) and time (MTTR) of the Node component. The operating mode is displayed through the Equation 6.

$$MO_{Node} = HW \wedge SO \wedge KVM \wedge NC \wedge VM \wedge VLC \wedge Apache \quad (6)$$

## CASE STUDIES

This section focus on three case studies for analysing the system availability; for applying sensitivity analysis and to application of redundancy in architecture from simulations obtained through the RBD and CTMC models. Case studies are distributed as follows:

i) Case Study I: availability analysis of baseline Architecture. ii) Case Study II: sensitivity analysis of all the system components (see **Figure 4**) to establish a ranking of the most important parameters in the video streaming service. iii) Case Study III: analysis the behavior of the system using the addition of redundancy to the architecture according to the results obtained by model simulations.

### Case Study I

**Figure 3** depicts the baseline architecture, which has the dedicated Frontend machine and another machine for the Node. **Table 1** compiles the MTTF and MTTR rates for the Frontend, Node, Volume and Service elements of this nonredundant architecture. Further details about this architecture are available in (Melo et al., 2014). These values were obtained from (Dantas et al., 2012; Bezerra et al., 2014; Kim et al., 2009).

**Table 1.** Parameters for RBD of the system

| Component | MTTF | MTTR |
|---|---|---|
| Frontend | 180.72 h | 0.96999 h |
| Node | 481.83 h | 0.91000 h |
| Volume | 100000 h | 1 h |
| Service | 217.77 h | 0.92633 h |

For the presented configuration of parameters, we find a value of 0.9885713 for the availability of the video streaming system, without redundancy. This availability corresponds to about 100 hours of downtime in a year, and therefore highlights the importance of searching for effective solutions to improve this system.

### Case Study II

In **Figure 4**, there is one Frontend and two Nodes, representing a redundant system. This system architecture is modeled in 6. The estrategies of sensitivity analysis, DASI and PDS were performed on these models to identify the most critical components of this Video Streaming Service architecture. The sensitivity indices were computed and were in **Table 2**.

**Table 2.** Ranking of sensitivity obtained through the DASI and PDS strategies in the Baseline model

| Parameter | $DASI^{*}_{\theta i}$ (\|As\|) | Parameters | $PDS^{*}_{\theta i}$ (\|As\|) |
|---|---|---|---|
| $\mu n$ | 0.098386059 | $\lambda n$ | $1.8850725\times10^{-3}$ |
| $\mu f$ | 0.005861235 | $\lambda f$ | $1.5866281\times10^{-5}$ |
| $\lambda f$ | 0.005331803 | $\mu f$ | $1.5866281\times10^{-3}$ |
| $\lambda vlc$ | 0.00296418 | $\lambda vlc$ | $8.8157368\times10^{-6}$ |
| $\mu vlc$ | 0.002754658 | $\mu vlc$ | $8.8157368\times10^{-6}$ |
| $\lambda n$ | 0.001884206 | $\mu n$ | $5.5918029\times10^{-6}$ |
| $\mu ap$ | 0.001391681 | $\lambda ap$ | $3.7634921\times10^{-6}$ |
| $\lambda ap$ | 0.001265953 | $\mu ap$ | $3.7621848\times10^{-6}$ |
| $\mu vol$ | 0.000011290 | $\mu vol$ | $2.9719022\times10^{-8}$ |
| $\lambda vol$ | 0.0000099999 | $\lambda vol$ | $2.9719022\times10^{-8}$ |
| $\lambda vm$ | 0.0000051851 | $\mu in$ | $1.9777666\times10^{-8}$ |
| $\mu in$ | 0.0000001542 | $\lambda vm$ | $1.5409453\times10^{-8}$ |

We identified at the top of the sensitivity analysis ranking for the DASI strategy, the components $\mu_n$, $\mu_f$, $\lambda_f$, $\lambda_{vlc}$, as the main candidate components for system improvement actions. This strategy gives us the possibility of not calculating the sensitivity index for a single configuration parameter but allows us the flexibility of the value variation. For PDS strategy identified at the top of the sensitivity analysis ranking, the components $\lambda_n$, $\lambda_f$, $\mu_f$, $\lambda_{vlc}$.

Comparing the DASI strategy in the **Table 2**, with the results obtained from the sensitivity index of the PDS strategy, presented in **Table 2**. We have identified that the ranking obtained by the DASI strategy is similar to that obtained by the PDS. This indicates that the DASI strategy obtained a coherent result regarding the parametric differential sensitivity strategy.

Thus, this indicates that the change made in the strategy provided satisfactory results. The results are equivalent, however, in the DASI strategy the repair rate of components appear before the failure rate in most cases. This is an interesting event because this time is small and most of the time we have only one maintenance team to carry out maintenance throughout the infrastructure.

### Case Study III

From the sensitivity analysis applied in the previous item, we identified that the Node is the most important component of the architecture of **Figure 3**, so we decided to apply redundancy in the cold standby mode, adding another Node to the infrastructure (see **Figure 4).** In **Table 3**, we have the values used for the models of **Figures 6, 7, 8** and **9**.

**Table 3.** Input parameters for the RBD cold standby model

| Module | Component | MTTF | MTTR |
|---|---|---|---|
| Frontend | HW | 8760 h | 100 min |
| | SO | 2895 h | 1 h |
| | CLC | 788.4 h | 1 h |
| | CC | 788.4 h | 1 h |
| | SC | 788.4 h | 1 h |
| | Walrus | 788.4 h | 1 h |
| Node | KVM | 2990 h | 1 h |
| | NC | 788.4 h | 1 h |
| | HW | 8760 h | 100 h |
| | SO | 2895 h | 1 h |
| | VM | 2880 h | 0.019166 h |
| | VLC | 788.4 h | 1 h |
| | APACHE | 336 h | 1 h |
| Volume | Volume | 100000 h | 1 h |
| SSMARKOV | SSMARKOV | 81.97 h | 0.4470 h |

**Table 4.** Consolidated input parameters for the cold standby model

| Parameters | Description | Values |
|---|---|---|
| $\lambda n$ | Failure rate of the *Node* | 1/150.0142 |
| $\mu n$ | Repair rate of the *Node* | 1/0.9232 |
| $\lambda f$ | Failure rate of the *frontend* | 1/180.72 |
| $\mu f$ | Repair rate of the *frontend* | 1/0.96999 |
| $\delta$ | Rate of activation of the secundary *Node* | 1/0.01667 |

After applying this cold standby redundancy mechanism, we left an availability of 0.988571 and a downtime of 100.12 hours for an availability of 0.994559 representing a downtime of 47.66 hours, this means a reduction in downtime of 47%.

## CONCLUSION

In this paper, we proposed analytical models to evaluate system availability of the VoD service. DASI and PDS strategies of sensitivity analysis were used to identify the most important parameters of the VoD service, guiding for availability and performance improvements. The results of this research identified that baseline architecture has an availability of 0.988571, which represents an annual downtime of 100 hours. After applying the cold standby redundancy mechanism, we found an availability of 0.994559. For future work, the authors propose to implement further redundancy mechanisms as warm standby and active-active models.and also to extend the scope of the models considered in this paper to incorporate network availability with different topologies, as well as other possible scenarios.

## REFERENCES


Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. et al. (2010). A view of cloud computing. *Communications of the ACM*, 53(4), 50–58. https://doi.org/10.1145/1721654.1721672

Bauer, E., Adams, R. and Eustace, D. (2011). *Beyond redundancy: how geographic redundancy can improve service availability and reliability of computer-based systems*. John Wiley & Sons. https://doi.org/10.1002/9781118104910

Bezerra, M. C., Melo, R., Dantas, J., Maciel, P. and Vieira, F. (2014). *Availability modeling and analysis of a vod service for eucalyptus platform*.

Chuob, S., Pokharel, M. and Park, J. S. (2011). Modeling and analysis of cloud computing availability based on eucalyptus platform for e-government data center. In *IEEE Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS),* 289–296. https://doi.org/10.1109/IMIS.2011.135

Dantas, J., Matos, R., Araujo, J. and Maciel, P. (2012). An availability model for eucalyptus platform: An analysis of warm-standy replication mechanism. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC),* 1664–1669. https://doi.org/10.1109/ICSMC.2012.6377976

Eucalyptus cloud computing platform - administrator guide. (2010). *Technical report*, Eucalyptus Systems, Inc., Version 2.0.

Hamby, D. M. (1994). A review of techniques for parameter sensitivity analysis of environmental models. *Environmental Monitoring and Assessment*, 32(2), 135–154. https://doi.org/10.1007/BF00547132

Khazaei, H., Misic, J., Misic, V. B. and Mohammadi, N. B. (2012). Availability analysis of cloud computing centers. In *IEEE Global Communications Conference (GLOBECOM),* 1957–1962. https://doi.org/10.1109/GLOCOM.2012.6503402

Kim, D. S., Machida, F. and Trivedi, K. S. (2009). Availability modeling and analysis of a virtualized system. In *IEEE 15th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC'09),* 365–371. https://doi.org/10.1109/PRDC.2009.64

Laprie, J.-C. (1995). Dependable computing and fault tolerance: Concepts terminology. In *IEEE International Symposium on Fault-Tolerant Computing,* 2–11. https://doi.org/10.1109/FTCSH.1995.532603

Maciel, P., Trivedi, K. S., Matias, R. and Kim, D. S. (2011). Dependability modeling. In *Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions*. Hershey: IGI Global.

Matos, R., Maciel, P. R. M., Machida, F., Kim, D. S. and Trivedi, K. S. (2012). Sensitivity analysis of server virtualized system availability. *IEEE Transactions on Reliability,* 61(4), 994–1006. https://doi.org/10.1109/TR.2012.2220711

Matos, R., Araujo, J., Oliveira, D., Maciel, P. and Trivedi, K. (2014). Sensitivity analysis of a hierarchical model of mobile cloud computing. *Simulation Modelling Practice and Theory*.

Melo, R., Clara, M., Dantas, J., Matos, R., Melo Filho, I. J. and Maciel, P. (2014). *Redundant vod streaming service in a private cloud: Availability modeling and sensitivity analysis*. Hindawi Publishing Corporation.

Melo, R., Marques Sobrinho, V. P. F., Melo Filho, I. J., Oliveira, A., Feliciano, F. and Maciel, P. R. M. (2017). Redundancy Mechanisms Applied in Cloud Computing infrastructures. In *13th Iberian Conference on Information Systems and Technologies (CISTI*), 1-6. https://doi.org/10.23919/CISTI.2017.7975950

Pannell, D. J. (1997). Sensitivity analysis of normative economic models: theoretical framework and practical strategies. *Agricultural economics*, 16(2), 139–152. https://doi.org/10.1016/S0169-5150(96)01217-0


 

Schaffer, S. (1994). *Babbage's Intelligence: Calculating Engines and the Factory System. Critical Inquiry.* The University of Chicago Press.

Silva, B., Callou, G., Tavares, E. A. G., Maciel, P., de Figueiredo, J. C., Souza, E., Araujo, C. J. M., Magnani, F. and Neves, F. A. S. (2012). Astro: An integrated environment for dependability and sustainability evaluation. *Sustainable Computing: Informatics and Systems*, 2, 1–31.

Yefremov, A. (2005). Failure-repair processes simulation for parallel redundant configurations using stochastic petri nets. In IEEE *9th Russian-Korean International Symposium on Science and Technology* (KORUS), 738–740. https://doi.org/10.1109/KORUS.2005.1507890