Journal of Computer and Communications, 2015, 3, 33-40
Published Online August 2015 in SciRes. http://www.scirp.org/journal/jcc
http://dx.doi.org/10.4236/jcc.2015.38004

Optimised Migrate Virtual Machine Rejuvenation

Sanheji Manel[1], Azizi Ridha[2], Maaloul Alia[1]

[1]Institute of Higher Technological Studies of Gabes, Gabes, Tunisia
[2]Institute of Higher Technological Studies of Sousse, Sousse, Tunisia
Email: sanhegi.manel@gmail.com, azizi_ridha@yahoo.fr, maaloul.alia@gmail.com

Received 15 June 2015; accepted 11 August 2015; published 14 August 2015

Abstract

Server virtualization is an essential component in virtualized software infrastructure such as cloud computing. Virtual machines are generated through a software called virtual machine monitor (VMM) running on physical servers. The risks of software aging caused by aging-related bugs affect both VM and VMM. As a result, service reliability degrades may generate huge financial losses to companies. This paper presents an analytic model using stochastic reward nets for time-based rejuvenation techniques of VMM and VM. We propose to manipulate the VM behavior while the VMM rejuvenation is according to the load on the system. Using a previous Petri net model of virtualized server, we performed an algorithm in order to optimize rejuvenation technique and achieve high availability. So we perform Migrate-VM rejuvenation or Warm-VM rejuvenation while there are current jobs in the system. Although Migrate-VM rejuvenation is better than Warm-VM rejuvenation in steady state availability, it can't be always performed as it depends on the capacity of the other host. When the queue is empty and the virtual machine has no current jobs to serve, we propose to combine both VMM rejuvenation and VM rejuvenation. We show that the proposed technique can enhance the availability of VMs.

Keywords

Component, Component, Availability, Data Center, Rejuvenation, Virtual Machine, Petri Net

1. Introduction

Software rejuvenation on virtualized environment is the emerging research area [1]. Server virtualization technology is one of today's essential components in data center. It allows generating multiple execution environments from hardware resources. By using software called virtual machine monitor (VMM), we create several

How to cite this paper: Manel, S., Ridha, A. and Alia, M. (2015) Optimised Migrate Virtual Machine Rejuvenation. *Journal of Computer and Communications*, **3**, 33-40. http://dx.doi.org/10.4236/jcc.2015.38004

virtual machines on top. Recently, cloud computing is using server virtualization in order to provide its service in internet. As the significant growth of the use of server virtualization, availability management of server virtualized system became an important issue.

For long running software, a restart of system components can improve the performance or availability of the system known as software rejuvenation. This technique was first introduced in order to heal software aging. However such technique may affect the availability of reliability of running server on virtual environment. Currently many companies are moving from physical hardware usage to cloud services. The restart of the software environment should be well chosen as it may cause cloud provider to suffer from huge loss of money.

Basically, rejuvenation can be triggered using either time policy or both load or time policy. Three rejuvenation techniques were highlighted. Cold-VM rejuvenation simply shuts down hosted VMs when the VMM is rejuvenating. Warm-VM rejuvenation suspends VMs before the VMM rejuvenation resumes the VMs after the rejuvenation. Migrate-VM rejuvenation moves the hosted VMs onto the other hosting server during the VMM rejuvenation by using live VM migration. VM migration may be used to improve VMM rejuvenation: all VMs on a host which presents VMM aging are moved to one host where aging effects were cleared [2].

In this paper, we first present a review of literature of rejuvenation technique. Many previous works have been done in the matter. Each one studied software rejuvenation differently. So we will discuss further the various rejuvenation techniques that have already been proposed. From previous work we conclude that system utilization load is an essential element when activating rejuvenation. So we present a comprehensive availability model for a server virtualized system of both VM and VMM using SPN. Achieving high availability in virtualized system is currently based on live Migration. However, migrating a VM to another host always depends on the capacity of the target host. Hence this technique can't always be deployed. So we will optimize the rejuvenation trigger for the VM considering the load on the system. We propose a new rejuvenation scheduling technique which rejuvenates aged VMs simultaneously with VMM rejuvenation when there are no current jobs on the virtual machine. The two objectives of our rejuvenation scheduling technique are achieving high availability and minimizing the job losses due to VMM rejuvenation.

The rest of the paper is organized as follows. Section II presents an overview on rejuvenation literature. Section III introduces the proposed rejuvenation scheduling technique for VMM with VMs. Section V shows the results of the simulation study which represent the effectiveness of the proposed approach. Section VI gives our conclusion.

2. Review of Literature of Rejuvenation Technique

Software rejuvenation was first introduced by Huang *et al*. [3] as a solution to mitigate software aging. The main idea is to restart in order to clear aging status. Since, many different policies have been proposed to implement software rejuvenation. In "Modeling and Analysis of Load and Time Dependent Software", a quantitative analysis of two software rejuvenation policies was presented. The first one considers only the ageing behavior of the system by time, while the second one considers the actual load of the system as well. Using SPN, they showed that significant performance gain can be obtained when the number of customers in the system is considered at rejuvenation [4].

Salfner and Wolter [5] present a queuing model to investigate the effect of time-based system rejuvenation on service availability using Petri Net. They investigated the effect of the frequency of rejuvenation on service availability. They modeled a finite queue to describe the service behavior while performing rejuvenation. The result shows that system utilization has a significant impact on service availability. Hence the optimal rejuvenation depends significantly on system utilization.

Machida, Kim and Trivedi presented an analytique model of software rejuvenation techniques of both VMM/VM using Petri Net [6]. They proposed comprehensive availability models of three rejuvenation techniques cold warm and Migrate-VM rejuvenation.

They defined three state of both VMM/VM. At first the machine is up. After running a while the machine becomes degraded due to software aging. If no rejuvenation is triggered the machine might go down and need other repair procedures.

In order to achieve higher steady-state availability of the VM, the rejuvenation trigger intervals of VM and VMM was determined in all three rejuvenation technique. The results shows that Migrate-VM rejuvenation helps achieve higher steady state availability. However the Warm-VM rejuvenation is not always better than

Cold-VM rejuvenation. This was explained by the fact that Warm VM rejuvenation does not clear the software aging when performing VMM rejuvenation by opposite to cold VM rejuvenation.

Machida [7] presented a cluster of servers and hosted virtual machines in data center. As the VMM rejuvenating might affects all the hosted VM's, the authors proposed that VM and VMM rejuvenate simultaneously. The proposed technique aims to increase VMs availability. To successfully migrate VM on another host, the VMM target should have enough free capacity. Since it's not always the case, Machida suggests migrating the degraded VM to rejuvenate simultaneously with another degraded VMM. This can reduce the downtime considerably and clear the aging bugs from both VMM and VM. The experimental results show that the proposed technique improves the VM availability. However to perform the suggested technique, VMM and VM needs to belong to the same owner.

Wang *et al*. [8] constructed a DSPN models for the cluster system with different rejuvenation policies. In policy-A is time based rejuvenation. Every period of time the rejuvenation will be carried out. The second policy-B (delayed rejuvenation), all nodes are merely scheduled for rejuvenation however in peak period the rejuvenation will be postponing until the off peak period. Policy-C is the combination of policy-A and policy-B. Rejuvenation is carried out only early in peak period; otherwise it will be delayed until the next off-peak period. Policy-C achieves the best throughput, while Policy-A achieves the best system availability. Policy-B is likely to outperform policy-A under optimal rejuvenation-triggering interval in terms of the expected system throughput.

Paing and Thein [9] presented optimization of resource usage on virtualized environment by accepting several services requests using stochastic Petri nets model under time-based rejuvenation policy for VMM. The goal was to analyze the availability of virtualized server system and resource usage management. When the VMM degrade, the rejuvenation manager of management server will trigger a rejuvenation operation. All the new requests and sessions are migrated from the virtual machine of the aging affected physical machine to virtual machine of other physical machine in the resource pool. When the ongoing requests are finished in aging infected PM, these VMM will be rejuvenated. The obtained results show that the use of virtualization, clustering technology and software rejuvenation mechanism can provide a very fast recovery to cut down the mean time to recovery to the minimum. It can achieve minimize downtime even in case of service restart.

3. The Model

3.1. Our Approach

During the VMM rejuvenation, we need to consider the actual state of the VM. If there are no jobs on the VM, it can simultaneously rejuvenate along with the VMM. If there is a load on the VM, VM's will either migrate to another host or suspend their current processes. Therefore, the proposed algorithm is proposed for guaranteeing a higher availability of VM.

Algoritm
A <- No traffic on the VM
B<- Traffic on the VM
C <-VMM rejuvenation
D <- Place is available in other VMM
Begin
If (A & C)
Then
all VM prone rejuvenate with VMM
and VM up stop
If (B & C)
Then (if D== True then migrate
Else suspend UP VM)
End

3.2. The Model

In **Figure 1** we introduce a DSPN model of a virtualized server with a preventive maintenance (rejuvenation) policy.

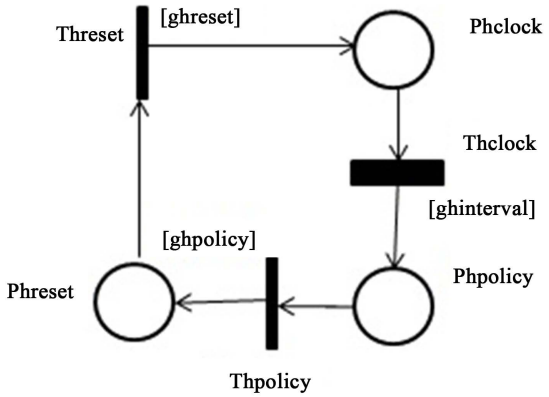

Figure 1. VMM clock.

3.3. Description:

Figure 2 shows the SRN for a server virtualized system with Optimized-VM rejuvenation. The model consists of four submodels; VMM model, VMM clock model, VM model, VM clock model and M/M/1/K queuing model.

The two clock models are used for triggering time-based rejuvenation of VM and VMM. A deterministic transition Tvinterval fires with duration 1/Tv after the latest boot time and starts the rejuvenation as long as the immediate transition Tvpolicy is enabled. When the VM rejuvenation process finishes, the immediate transition Tvreset is enabled and a token is deposited in the place Pvclock again. Similarly, Thinterval is a deterministic transition for triggering VMM rejuvenation with duration 1/Th after the VMM startup time.

The DSPN of **Figure 2** represents the M/M/1/K queuing model for the server. The server capacity is modeled with the place free. The transition Tarrival models the arrival of jobs. The number of jobs enqueued in the server is modeled by the place buffer models. The transition Tservice represents the service event. Tempty is enabled when the place the Virtual machine is not available.

Figure 3 models the VMM model. The Phup transition represents the robust state of the VMM. Later when the transition Thfp fires, a token is deposited in Phfp and represents the software aging of the VMM. If the transition Thfail fires, a token is deposited in Phfail which represents the VMM failure due to the software aging. The VMM failure can be detected by the monitoring tool at a certain monitoring interval and is manually recovered by a system administrator which is modeled by firing of Thdet. Threpair fires and a token is deposit in Phup. When the VMM rejuvenation is triggered by the VMM clock model, the immediate transition either Threjt or Thfprejt is enabled and a token is deposited in Phrej. When the VMM is finish rejuvenating and all the aging states are cleared, Threj fires and a token is deposited in Phup.

The VM behavior is similar to VMM's. At first the VM is up modeled by transitions Pvup. Later the VM is degraded which represented by the firing of transition Tvfp to place Pvfp. If VM rejuvenation is triggered by the VM clock, Thfprej fires and a token is deposit in Pvrej. Due to software aging the VM may go down, then Tvfail is fired and a token is deposit in place Pvfail (**Figure 4** and **Figure 5**).

When the VMM rejuvenation is triggered, the Optimised-rejuvenation makes hosted VM either migrate on other host or suspend operation to stop the execution of their current jobs before starting the VMM rejuvenation. This policy is represented in the guard functions for the immediate transitions Tvprem, Tvpres Tvfprem and Tvfppres. Those immediates transitions fires once are a token is deposited in Phpolicy in the VMM clock model. If place Buffer is empty, then the VM is not busy and can rejuvenate simultaneously along with the VMM. When the VM is serving jobs, it will first look for a free capacity on other host. In that case, a token is deposit in Pvmig or Pvmigd or Pvfpmig or Pvfpmigd by firing Tvprem or Tvfpprem, the VMM can start the rejuvenation. When the rejuvenation is completed, the immediate transition Tvpost and Tvfppost are enabled and the VM returns back to the original host by live VM migration. When there is no free capacity on another host, the VM suspends operation at the VMM rejuvenation. Tvpres and Tvfppres are enabled and a token is placed in Pvsus or Pvfpsus. When a token is deposited in Pvsusd or Pvfpsusd by firing Tvsus or Tvfpsus, the VMM can start the rejuvenation. The immediate transition Tvpost and Tvfppost are enabled, when the VMM rejuvenation is com-

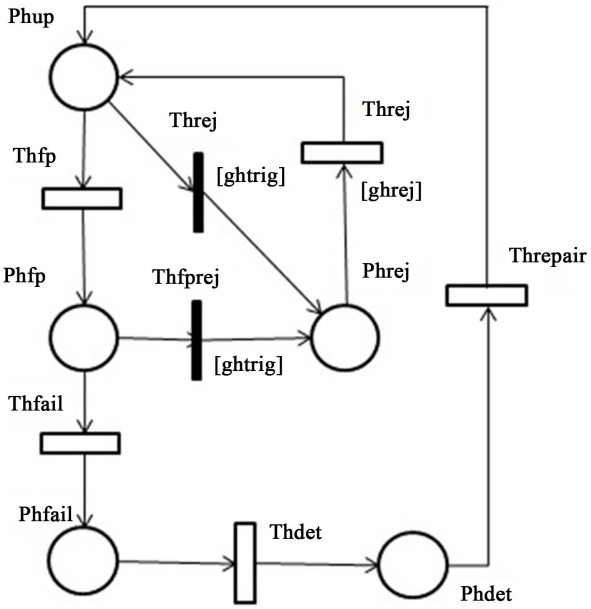

Figure 2. VMM model.

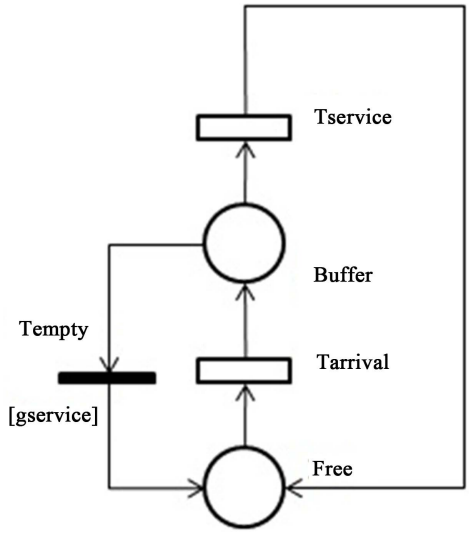

Figure 3. M/M/1/K queueing model.

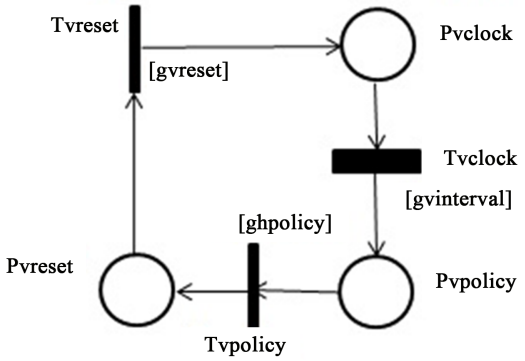

Figure 4. VM clock.

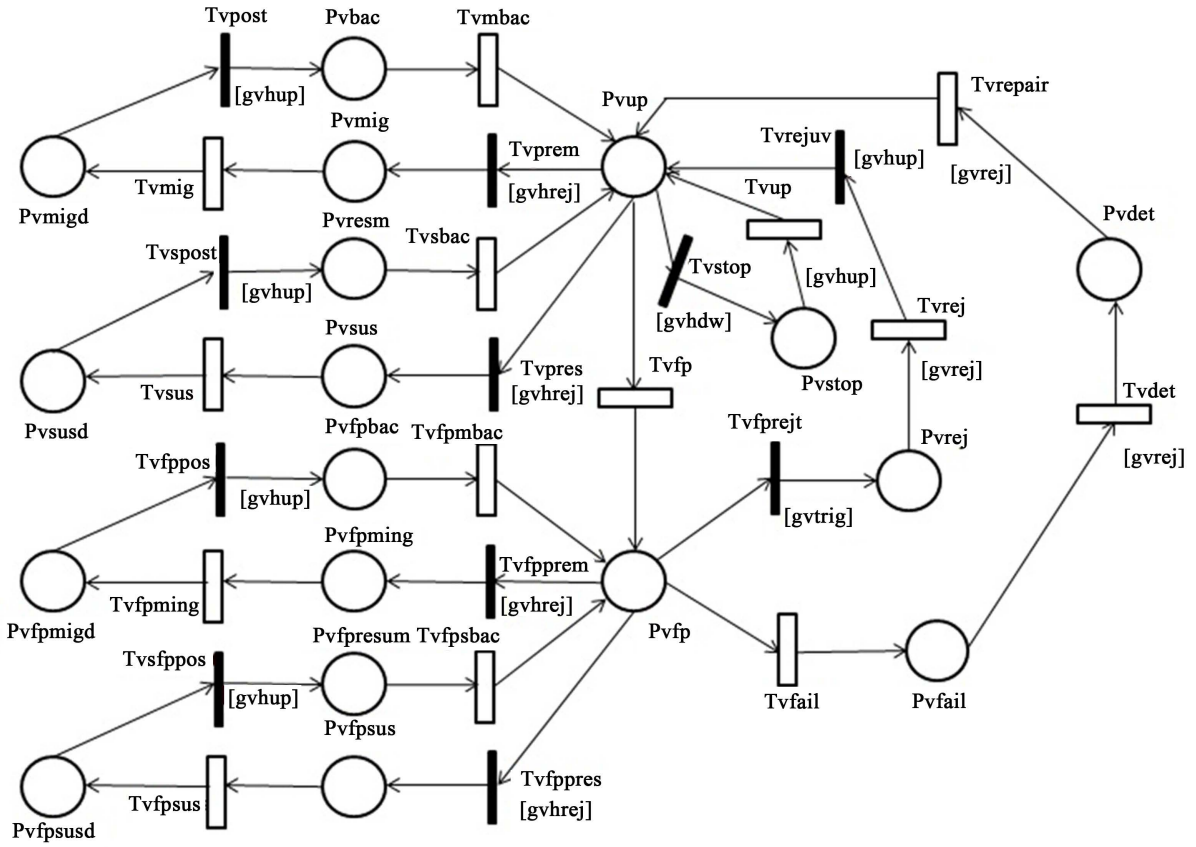

Figure 5. VM model.

pleted and a token is deposited in Phup in the VMM model. The hosted VM resumes the execution when the token is deposited in Pvup or Pvfp by the transitions Tvresm or Tvfpresm. We assume that the VMM rejuvenation can start unless a token is deposited in Pvup, Pvfp, Pvsus, Pvfpsus, Pvresm or Pvfpresm (see **Table 1**).

4. Numerical Results

Three rejuvenation techniques are compared by numerical solution of SRNs using SPNica. SPNica is based on Mathematica [10].

System Availability

Under the given parameter values, we compute steady-state availability by varying the rejuvenation trigger intervals of VM and VMM. Frequent rejuvenation increases the down time. Moreover, less frequent rejuvenation also increases the down time caused by software failure. The optimum rejuvenation trigger interval is the combination of both VM and VMM rejuvenation trigger that maximize VM's steady-state availability. As a result (**Table 2**), Ms-VM rejuvenation outperforms the other three rejuvenation techniques in terms of steady-state availability. Since the VMM rejuvenation has negligible effect on the VM availability, VMM rejuvenation can be performed more frequently than VM rejuvenation (**Table 3**).

5. Conclusions

In this paper, we presented a comprehensive availability model based on SPN to improve system availability in virtualized environment. The models enable the choice of the appropriate rejuvenation policy according to the traffic on the system. The proposed technique combines the three rejuvenation techniques and each time according to the load on the system, it decides which one to perform. When there is no load on the system, the VM re-

Table 1. Guard functions.

Guard functions	
ghintervall	(#Phup==1)||(#Phfp==1)
ghpolicy	(#Pvmigd==1)||(#Pvfpmigd==1)||(#Pvfail==1)||(#Pvdet==1)||(#Pvrej==1)||(#Pvsus==1)||(#Pvfpsus==1)
ghreset	#Phrej==1
gvintervall	(#Pvmigd==1)||(#Pvfpmigd==1)||(#Pvfail==1)||(#Pvdet==1)||(#Pvrej==1)||(#Pvsus==1)||(#Pvfpsus==1)
gvpolicy	(#Pvup==1)||(#Pvfp==1)||(#Pload==0)
gvreset	#Pvrej==1
ghtrig	#Phtrigger==1
gvtrig	(#Pvtrigger==1)||(Pload==0)
gvhup	(#Phup==1)||(#Phfp==1)
gvhrej	(#Phpolicy==1) && (#Pload==1)
ghrej	#Phclock==1
gvrej	#Pvclock==1 && ((#Phup==1)||(#Phfp==1))
gvhdw	(#Phfail==1)((#Phpolicy==1) && (#Pload==0)
gservice	(#Pvup==1)||(#Pvfp=1)

Table 2. Parameters used in the models.

Transitions	Mean values
VM aging	1 week
VM failure rate after aging	3 days
VM failure detection rate	5 min
VM failure recovery rate	30 min
VM rejuvenation rate	1 min
VM suspend rate	0.08 sec
VM resume rate	0.8
VM migration rate	1 sec
VM rejuvenation trigger rate (T_V)	1 day
VMM aging rate	1 month
VMM failure rate after aging	1 week
VMM failure detection rate	5 mins
VMM failure recovery rate	1 hour
VMM rejuvenation rate	2 mins
VMM rejuvenation trigger rate T_h	1 week

Table 3. Optimum rejuvenation schedules.

Methods	$1/T_V$	$1/T_h$	Steady state availability
Cold	34.72	121.95	0.99808
Warm	31.95	112.36	0.99811
Migrate	31.95	10.95	0.99871
MS-VM	31.95	10.95	0.99911

juvenation is planned simultaneously along with VMM. Thus we reduce the unnecessary shutdowns of robust VMs and hence improve the VM availability. The results show that the proposed technique achieves higher availability than previous techniques. This technique is potentially a suitable rejuvenation technique for virtualized data center, achieving significant improvement in the steady-state availability.

In future works, we intend to focus on live migration mechanism in cloud computing environment with time-based rejuvenation under varying workload. Using more scenarios with different types of workloads may show the effect of migration on system availability.

References

[1] Umesh, I.M., Srinivasan, G.N. and Cholli, N.G. (2014) A Study on Software Rejuvenation Techniques on Virtualized Environment. *International Journal of Science*, *Engineering and Technology Research* (*IJSETR*), **3**, 2110-2112.

[2] Melo, M., Araujo, J., Matos, R., Menezes, J. and Maciel, P. (2013) Comparative Analysis of Migration-Based Rejuvenation Schedules on Cloud Availability. *IEEE International Conference on Systems*, *Man*, *and Cybernetics*, Manchester, October 2013, 4110-4115.

[3] Huang, Y., Kintala, C., Kolettis, N. and Fulton, N.D. (1995) Software Rejuvenation: Analysis, Module and Applications. *Proceedings of* 25*th Symposium on Fault Tolerant Computing*, Pasadena, 27-30 June 1995, 381-390. http://dx.doi.org/10.1109/ftcs.1995.466961

[4] Trivedi, K.S. (1996) Modeling and Analysis of Load and Time Dependent Software. *The Third International Workshop on Performability Modeling of Computer and Communication Systems*, Illinois, 6-8 September 1996.

[5] Salfner, F. and Wolter, K. (2008) A Queuing Model for Service Availability of Systems with Rejuvenation. *IEEE International Conference on Software Reliability Engineering Workshops*, Seattle, 11-14 November 2008, 1-5. http://dx.doi.org/10.1109/issrew.2008.5355520

[6] Machida, F., Kim, D.S. and Trivedi, K.S. (2010) Modeling and Analysis of Software Rejuvenation in a Server Virtualized System. *IEEE Second International Workshop on Software Aging and Rejuvenation* (*WoSAR*), San Jose, 2 November 2010, 1-6. http://dx.doi.org/10.1109/wosar.2010.5722098

[7] Machida, F., Xiang, J.W., Tadano, K. and Maeno, Y. (2012) Combined Server Rejuvenation in a Virtualized Data Center. 9*th International Conference on Autonomic & Trusted Computing* (*UIC/ATC*), Fukuoka, 4-7 September 2012, 486-493.

[8] Wang, D., Xie, W. and Trivedi, K.S. (2007) Performability Analysis of Clustered Systems with Rejuvenation under Varying Workload. *Performance Evaluation*, **64**, 247-265. http://dx.doi.org/10.1016/j.peva.2006.04.002

[9] Paing, A.M.M. and Thein, N.L. (2012) High Availability Solution: Resource Usage Management in Virtualized Software Aging. *International Journal of Computer Science and Telecommunications*, **3**, 1-10.

[10] German, R. (2000) Performance Analysis of Communication Systems—Modeling with Non-Markovian Stochastic Petri Nets. John Wiley & Sons Ltd., Hobokon.