2011 International Conference on Information Communication and Management
IPCSIT vol.16 (2011) © (2011) IACSIT Press, Singapore

A Petri Net Model for High Availability in Virtualized Local Disaster Recovery

Aye Myat Myat Paing and Ni Lar Thein

University of Computer Studies, Yangon

Abstract. Effective business continuity strategies need to have both high availability (HA) and disaster recovery (DR). Traditional high availability and disaster recovery solutions require a great deal of duplicate hardware and software. Virtualization affords significant cost and performance advantages over more traditional disaster recovery options. In this paper, the virtualized local disaster recovery is proposed in order to provide a higher level of availability for business continuity. Effective way to use the resources at both primary site and disaster recovery site, active-active clustering architecture is employed in this virtualized local DR. The software failure due to software aging problem which can decrease the availability of the system is considered. The software rejuvenation methodology and virtualization technology is combined to counteract the software aging problem. To get higher level of availability in local disaster recovery, virtualization, clustering and software rejuvenation technology are integrated. A stochastic Petri net model is constructed to describe the behavior of virtualized local DR. To analyze the availability of the proposed model, numerical derivation is presented. The numerical derivation results are validated with the evaluation through SHARPE simulation tool.

Keywords: availability, clustering, local disaster recovery, stochastic Petri nets, virtualization.

1. Introduction

Organizations today face a tough challenge in choosing an appropriate high availability solution in order to minimize the amount of downtime that meets their business requirements. As a result, disaster recovery has gained great importance in IT. There are two broad categories such as natural or man-made disaster which can cause site failures. The disaster recovery is designed to ensure the continuation of vital business processes in the event that the disaster occurs [1].

Availability has long been a critical issue for online computer systems whose failure can halt business processes [2]. The need for high availability (HA) and disaster recovery (DR) in IT environment is more importance than other sectors of enterprises. Traditional high availability and disaster recovery solutions require a great deal of duplicate hardware and software.

Virtualization technology is changing the face of disaster recovery. Virtual machine technologies require availability solutions that provide protection against data loss and downtime for the entire environment.

A cluster is a collection of computer nodes: independent, self-contained computer systems working together – to provide a more reliable and powerful system than a single node alone. Therefore, clustering is a key server function in HA and DR environment. Active-active clustering is a high availability (synchronous) DR solution. The architecture is maximizing the performance of the applications, and providing more cost-effective use of the resources. The benefit of shared storage is that the nodes simultaneous own the shared resources [3].

Disaster and its recovery processes involve unplanned interruption of service. Unplanned downtime is mainly caused by computer failure, network failure, software failure and local or regional disaster. As business becomes increasingly dependent on information and computing technology, continuous availability is a universal concern. Failures of computer systems are more often due to software faults than due to

hardware faults. The state of software degrades with time is known as software aging. The most effective way to handle software failure due to software aging is software rejuvenation. Software rejuvenation is a proactive fault management technique aimed at cleaning up the system internal state to prevent the occurrence of more severe crash failures in the future [4], [5].

In this paper, we combine clustering technology, virtualization technology and software rejuvenation mechanism in order to improve the DR performance. Analytical models are mathematical models which are an abstraction from the real world system and relate only to the behavior and characteristics of interest. We construct a Petri net model for virtualized local DR and evaluate through both analytical and SHARPE tool simulation.

2. Proposed Virtualized Local Disaster Recovery

Local disaster recovery is that the surviving node can support the service for a failed node in the event of localized disaster such as fire or building power outages.

The proposed virtualized local DR is based on active-active high-availability and DR solution working with virtual servers and shown in Figure 1. Clustering supports two or more servers running duplicate VMs. Failover technologies also allow a failed VM to load from a storage snapshot and start up on another server. Each service under high availability DR solution needs at least two sites: primary site and disaster recovery site. The active physical server at primary site as well as DR site contains two or more VMs. At primary site, VMs are created as active VMs and standby VMs. At the same time, VMs are created as active VMs and standby VMs. The active-standby roles are switched due to failure detection. In this way, hardware exposure is mitigated through physical hardware redundancy.

Clustering provides high availability by protecting against a node failure. However, it does not prevent against storage failures. Therefore we employ shared storage for this virtualized clustering architecture. Both sites are designed to operate concurrently under normal conditions and to serve as a DR site for each other, should a complete site failure occur at either one. A heartbeat keep- alive message is used to monitor the health of the nodes between primary site and DR site.

In this virtualized local disaster recovery, when such a disaster occurs, the virtual machines are transferred and reloaded on the servers at the alternate site. We also consider the software failure due to software aging problem which can decrease the availability of the system. In virtualized environments, the hypervisor itself or virtual machines can fail with software failure. The software rejuvenation methodology and virtualization technology is employed to counteract the software aging problem in the proposed local DR. The active VMs at primary site as well as DR site provide different services. When one of the active VMs at primary site needs to do rejuvenation, the services are migrated to one of the standby VMs at DR site. VMs can be failed over through migration or restarted from storage onto active server at DR site but the migration downtime is nearly zero and we can neglect the duration of migration. As a result, virtualization technology can provide continued services even if VMs need to perform rejuvenation.

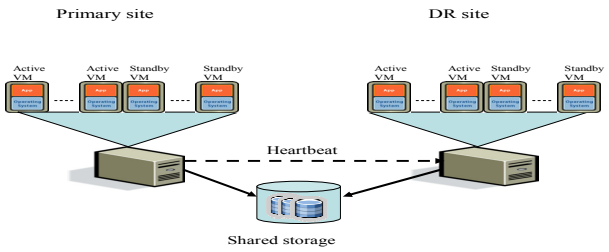


Fig.1: Active-Active Virtualized Clustering Architecture

3. Proposed Petri Net Model for Virtualized Local DR

Petri Nets (PN) are a graphical tool for the formal description of the flow of activities in complex systems. Petri nets are extended by associating time with the firing of transitions, resulting in Timed Petri

nets. A special case of Timed Petri nets is Stochastic Petri nets (SPN) where the firing times are considered to be random variables.

The proposed stochastic Petri net model of virtualized local DR is shown in Figure 2. Each physical server includes the following the failure and recovery transitions. The circles represent places with dots inside representing the tokens held inside that place. The robust and healthy state is modeled by the place P_h. It has n tokens which represented n virtual machines in active physical server. Transition T_{probf} models the aging of the software. When this transition fires,(i.e., a token reaches place P_{probf}) the VM enters the failure probably state. The transition T_f models crash failure of the VM. During the VM restarts from DR site while the transition T_{sw} is enabled, every other activity is suspended, the inhibitor arc from place P_{fail} to transition T_c is used to model this fact.

The transition T_c models the rejuvenation period. This transition is competitively enabled with T_{probf} and fires when the clock expires if T_{probf} has not fired by that time. Once its fires, token moves in the place P_{rej} and the activity related with software rejuvenation. During the rejuvenation, every other activity in the VM is suspended. This is modeled by inhibitor arcs from P_{rej} to transitions T_{probf} and T_f. Upon rejuvenation, the net has to be reinitialized into a condition with at least one token in the place P_h and one in the place P_{clock}, and all the other places empty. If the VM was in the robust state when T_c fired, then after rejuvenation is complete, T_{r2} fires to re-initialize the net. If the VM had reached the failure probably state (token in P_{probf}), then T_{r1} fires to complete the rejuvenation and reinitializes the net.

In the P_h state, failure can cause by hardware faults represented by the firing transition T_{hw}. Firing of transition T_{sw} represents the switch over where another active physical server from DR site takes overall the running operation of the fail physical server from primary site. Transition T_{sw} can only fire when the DR site available. After the primary site come back online, the DR site will put back all operations it obtained from primary site. This return action is represented by the firing of T_{swbk}. Unless services are migrated, the transition T_{Down} will fire and reach the fail state (P_{Down}). After that T_{repair} fires to repair the both physical hosts and return to healthy and operational state.

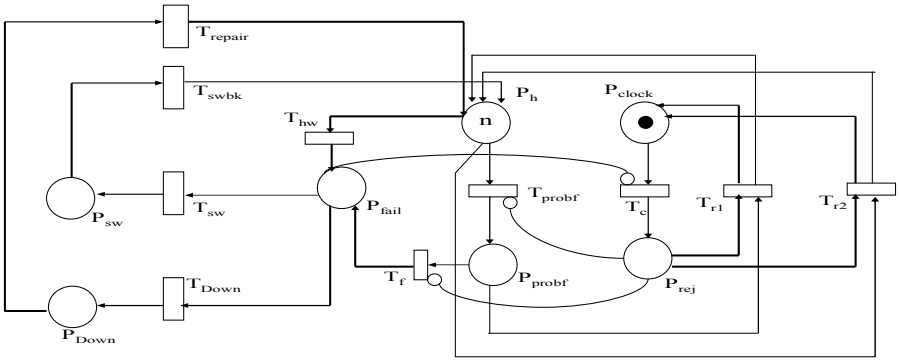


Fig.2: Stochastic Petri Net model for Virtualized LocalDR

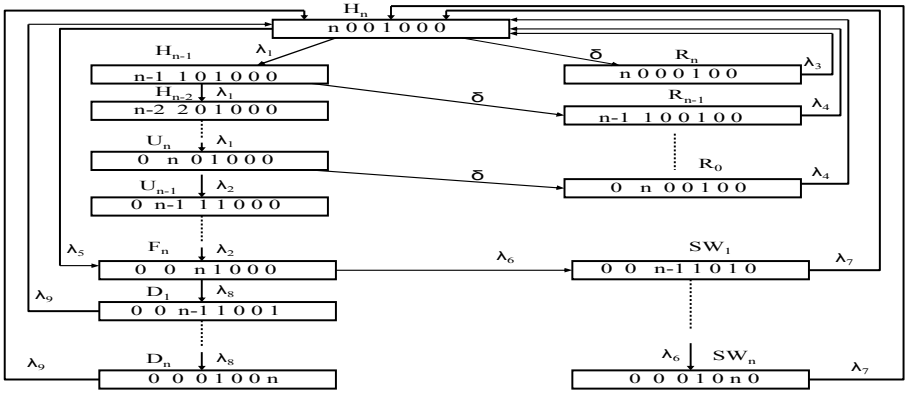
P_h : Healthy state P_{probf} : Failure Probably state P_{fail} : VMs or one active physical host failure state
P_{rej} : Rejuvenation state P_{clock} : Rejuvenation Interval state P_{sw} : Switchover state
P_{Down} : Both sites are in failure state

The assumptions for the proposed model are the distribution of time between hardware failures is assumed to be exponential. Time between software failures caused by aging-related faults is assumed to be hypo-exponentially distributed. Failure detection time and switchover, repair, rejuvenation, switchback time are assumed to be exponentially distributed.

3.1 Reachability graph

In this section, we construct the reachability graph for the proposed model. This graph represents the active physical host with n virtual machine servers at both primary and DR site as shown in Figure 3. Let 7 tuple (P_h, P_{probf}, P_{fail}, P_{clock} P_{rej}, P_{sw}, P_{Down}) denote the marking with P_x =1, if a token is presented in place P_x, and zero otherwise. A marking is reachable from another marking if there exists a sequence of transition

firings starting from the original marking that result in the new marking. The marking process is mapped into a continuous time Markov chain (CTMC) with state space isomorphic to the reachability graph of the PN.



Fig,3: Reachability Graph for the proposed model

Figure 3 illustrates the reachability graph with squares representing the markings and arcs representing possible transition between the markings. Let λ_1, λ_2, λ_3, λ_4, λ_5, λ_6, λ_7, λ_8 and λ_9 be the transition rates associated with T_{probf},T_f, T_{r2}, T_{r1},T_{hw}, T_{sw}, T_{swbk}, T_{Down} and T_{repair} respectively. And also, let δ be the firing time associated with transition T_c. By mapping through actions to this reachability graph with stochastic process, we get mathematical steady-state solution of the chain.

We may compute the steady-state probability by first writing down the steady-state balance equations of figure as follows.

For marking H_n (n= number of VMs)

$$\lambda_3 P_{R_n} + \lambda_4 \sum_{i=1}^{n} P_{R_{n-i}} + \lambda_7 \sum_{i=1}^{n} P_{SW_i} + \lambda_9 \sum_{i=1}^{n} P_{D_i} = \lambda_5 P_{H_n} + \lambda_1 P_{H_n} + \delta P_{H_n} \quad (1)$$

For marking H_{n-i} (n= number of VMs, i= 1 to n-1)

$$\lambda_1 P_{H_{(n-i)+1}} = \lambda_1 P_{H_{n-i}} + \delta P_{H_{n-i}} \quad (2)$$

For marking R_n (n= number of VMs)

$$\delta P_{H_n} = \lambda_3 P_{R_n} \quad (3)$$

For marking R_{n-i} (n= number of VMs, i= 1 to n-1)

$$\delta P_{H_{n-i}} = \lambda_4 P_{R_{n-i}} \quad (4)$$

For marking R_0 (n= number of VMs)

$$\delta P_{U_n} = \lambda_4 P_{R_0} \quad (5)$$

For marking U_n (i = n-1, n= number of VMs)

$$\lambda_1 P_{H_{n-i}} = \lambda_2 P_{U_n} + \delta P_{U_n} \quad (6)$$

For marking U_{n-i} (n= number of VMs, i= 1 to n-1)

$$\lambda_2 P_{U_{(n-i)+1}} = \lambda_2 P_{U_{n-i}} \quad (7)$$

For marking F_n (i = n-1, n= number of VMs)

$$\lambda_2 P_{U_{n-i}} + \lambda_5 P_{H_n} = \lambda_6 P_{F_n} + \lambda_8 P_{F_n} \quad (8)$$

For marking D_i (n = number of VMs, i= 1 to n-2)

$$\lambda_8 P_{F_n} = \lambda_8 P_{D_1} + \lambda_9 P_{D_1} \quad (9)$$

$$\lambda_8 P_{D_i} = \lambda_8 P_{D_{i+1}} + \lambda_9 P_{D_{i+1}} \quad (10)$$

For marking D_n (n = number of VMs)

$$\lambda_8 P_{D_{n-1}} = \lambda_9 P_{D_n} \quad (11)$$

For marking SW_i (n = number of VMs, i= 1 to n-2)

$$\lambda_6 P_{F_n} = \lambda_6 P_{SW_1} + \lambda_7 P_{SW_1} \quad (12)$$

$$\lambda_6 P_{SW_i} = \lambda_6 P_{SW_{i+1}} + \lambda_7 P_{SW_{i+1}} \quad (13)$$

For marking SW_n (n = number of VMs)

$$\lambda_6 P_{SW_{n-1}} = \lambda_7 P_{SW_n} \tag{14}$$

The conservation equation of figure is obtained by summing the probabilities of all states in the system and the sum of equation is 1.

$$\sum_{i=1}^{n} P_{H_i} + \sum_{i=0}^{n} P_{R_i} + \sum_{i=1}^{n} P_{U_i} + P_{F_n} + \sum_{i=1}^{n} P_{D_i} + \sum_{i=1}^{n} P_{SW_i} = 1 \tag{15}$$

Combining the above-mentioned balance equations with the conservation equation, and solving these simultaneous equations, we acquire the closed-form solution for the system.

$$P_{R_n} = \frac{\delta}{\lambda_3} P_{H_n} \tag{16}$$

(n= number of VMs)

$$P_{H_{n-i}} = \left[\frac{\lambda_1}{\lambda_1+\delta}\right]^i P_{H_n} \tag{17}$$

(n= number of VMs, i =1 to n-1)

$$P_{R_{n-i}} = \frac{\delta}{\lambda_4}\left[\frac{\lambda_1}{\lambda_1+\delta}\right]^i P_{H_n} \tag{18}$$

(n= number of VMs, i =1 to n-1)

$$P_{R_0} = \frac{\delta}{\lambda_4} A \left[\frac{\lambda_1}{\lambda_1+\delta}\right]^{n-1} P_{H_n} \tag{19}$$

(n= number of VMs)

$$P_{U_i} = A \left[\frac{\lambda_1}{\lambda_1+\delta}\right]^{n-1} P_{H_n} \tag{20}$$

(n= number of VMs, i =1 to n)

$$P_{F_n} = C P_{H_n} \tag{21}$$

(n= number of VMs)

$$P_{SW_i} = C \left[\frac{\lambda_6}{\lambda_6+\lambda_7}\right]^i P_{H_n} \tag{22}$$

(n= number of VMs, i =1 to n-1)

$$P_{D_i} = C \left[\frac{\lambda_8}{\lambda_8+\lambda_9}\right]^i P_{H_n} \tag{23}$$

(n= number of VMs, i =1 to n-1)

$$P_{SW_n} = \frac{\lambda_6}{\lambda_7} C \left[\frac{\lambda_6}{\lambda_6+\lambda_7}\right]^{n-1} P_{H_n} \tag{24}$$

(n= number of VMs)

$$P_{D_n} = \frac{\lambda_8}{\lambda_9} C \left[\frac{\lambda_8}{\lambda_8+\lambda_9}\right]^{n-1} P_{H_n} \tag{25}$$

(n= number of VMs)

$$P_{H_n} = \left\{1 + \frac{\delta}{\lambda_3} + \sum_{i=1}^{n-1}\left[\left[\frac{\lambda_1}{\lambda_1+\delta}\right]^i\right] + \frac{\delta}{\lambda_4}\left[\sum_{i=1}^{n-1}\left[\frac{\lambda_1}{\lambda_1+\delta}\right]^i\right] + \frac{\delta}{\lambda_4} A \left[\frac{\lambda_1}{\lambda_1+\delta}\right]^{n-1} + C + A\left[\sum_{i=1}^{n}\left[\frac{\lambda_1}{\lambda_1+\delta}\right]^{n-1}\right] + C\left[\sum_{i=1}^{n-1}\left[\frac{\lambda_6}{\lambda_6+\lambda_7}\right]^i\right] + \frac{\lambda_6}{\lambda_7} C \left[\frac{\lambda_6}{\lambda_6+\lambda_7}\right]^{n-1} + C\left[\sum_{i=1}^{n-1}\left[\frac{\lambda_8}{\lambda_8+\lambda_9}\right]^i\right] + \frac{\lambda_8}{\lambda_9} C \left[\frac{\lambda_8}{\lambda_8+\lambda_9}\right]^{n-1}\right\}^{-1} \tag{26}$$

Where $A = \left[\frac{\lambda_1}{\lambda_{2+\delta}}\right]$, $B = A\left[\frac{\lambda_1}{\lambda_1+\delta}\right]^{n-1}$, $C = \frac{1}{\lambda_6+\lambda_8}[\lambda_2 B + \lambda_5]$

The meaning of the probabilities as follows:

P_{H_i}	The probability of the VM is in healthy state
P_{R_i}	The probability of the VM is in rejuvenation state
P_{U_i}	The probability of the VM is in failure probably state
P_{F_n}	The probability of the VMs or one active physical host is in failure state
P_{D_i}	The probability of the both sites are in failure state

P_{SW_i} The probability of the VM is in switchover state
(i= 1 to n, n= number of VMs)

3.2 Availability and downtime in analysis

In the proposed model, services are not available when both primary site and DR site are down. We also define the availability of the proposed model as:

$$Availability = 1 - \sum_{i=1}^{n} P_{D_i} \tag{27}$$

The expected total downtime of the application with rejuvenation in an interval of T time units is

$$Downtime(T) = \sum_{i=1}^{n} P_{D_i} \times T \tag{28}$$

3.3 Numerical examples

In this section, we illustrate the applicability of the proposed model and solution methodology through numerical examples. The exact model transition firing rates for the mod model are not known, a good estimate value for a range of model transition firing rates is assumed. For this purpose, we perform experiments using the following failure profile mentioned in Table 1.

Table 1 Transition Firing Rates

Transition	Firing Rate (hr $^{-1}$)
T_{fail}	3times/month
T_{probf}	1 time /a day
T_{hw}	$1/10^6$
T_{repair}	2times/a day
$1/T_{r1}$	5 mins
$1/T_{r2}$	5 mins
$1/T_{sw}$	3 mins
$1/T_{swbk}$	3 mins
T_{Down}	3times/month
T_c	variable

Figure 4 illustrates the availability changes for the proposed model with different rejuvenation intervals and different number of VMs had been applied. Best rejuvenation intervals for VMs were 300 hours. It can be observed that the lower the rejuvenation interval rate, the higher availability can be achieved. When the number of VMs increased, the higher availability can be achieved. Figure 5 plotted the downtime as a function of the rejuvenation interval for different number of VMs. The downtime reaches the minimum at the rejuvenation interval at 300 hours.

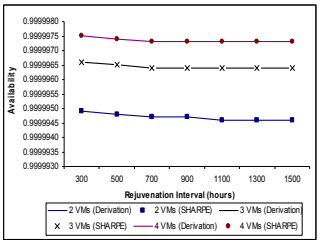


Fig.4: Availability vs. different rejuvenation interval and different number of VMs

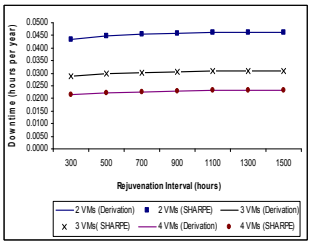


Fig.5: Downtime vs. different rejuvenation interval and different number of VMs

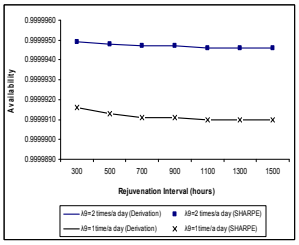


Fig.6: Availability vs. different rejuvenation interval and different repair rates

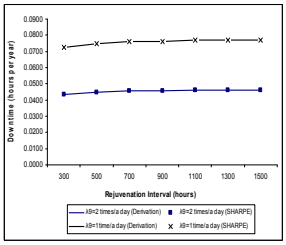


Fig.7: Downtime vs. different rejuvenation interval and repair rates

The influence of different repair transition firing rates and rejuvenation intervals on availability is shown in Figure 6. The repair transition firing rates are assumed 2times/a day and 1time/a day for 2 VMs. The faster repair, the higher availability of the proposed model we will get. Therefore, the availability is dependent on the repair transition rate when both sites are down. The figure 7 shows the differences in downtime with different rejuvenation intervals and different repair transition firing rates. From the result, it is apparent that the proposed model is a high availability in order to integrate virtualization technology, clustering and

software rejuvenation mechanism. According to the figures it is found that the derivation results and SHARPE tool simulation results are the same.

4. Conclusion

Many businesses require the availability of business-critical applications 24 hours a day, seven days a week and can afford no data loss in the event of disaster. To implement this requirement, organizations must give high availability and disaster recovery. High availability systems require fewer failure and faster repair. In this paper, active-active clustering architecture is presented in virtualized local disaster recovery in order to fully utilize resources and which can provide a high level of availability for business continuity. A Petri net model of virtualized local DR is presented and expressed availability and downtime in terms of the transition firing rates in the model. The numerical results are validated with the evaluation through SHRPE tool. It is found that the derivation results and the SHARPE result are the same. The obtained results showed that combining virtualization technology and software rejuvenation methodology can improve DR performance.

5. References

[1] B. C. Martin, *Disaster Recovery Plan Strategies and Processes*, Version 1.3, February 2002.

[2] E.Vargas, *High Availability Fundamentals*, Sun Blueprints Series, 2000. Online Available: http://www.sun.com/blueprints.

[3] http://www.citrix.com/

[4] Software Rejuvenation. Department of Electrical and Computer Engineering, Duke University Online Available: http://www.software-rejuvenation.com/.

[5] Y. Huang, C. Kintala, N. Kolettis and N. D. Fultion, *Software Rejuvenation: Analysis, Module and Applications*, June 1995.