World Academy of Science, Engineering and Technology
International Journal of Electrical and Computer Engineering
Vol:16, No:3, 2022

A Timed and Colored Petri Nets for Modeling and Verifying Cloud System Elasticity

W. Louhichi, M.Berrima, N. Ben Rajeb Robbana

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

***Abstract*—**Elasticity is the essential property of cloud computing. As the name suggests, it constitutes the ability of a cloud system to adjust resource provisioning in relation to fluctuating workloads. There are two types of elasticity operations, vertical and horizontal. In this work, we are interested in horizontal scaling, which is ensured by two mechanisms; scaling in and scaling out. Following the sizing of the system, we can adopt scaling in the event of over-supply and scaling out in the event of under-supply. In this paper, we propose a formal model, based on temporized and colored Petri nets (TdCPNs), for the modeling of the duplication and the removal of a virtual machine from a server. This model is based on formal Petri Nets (PNs) modeling language. The proposed models are edited, verified, and simulated with two examples implemented in colored Petri nets (CPNs)tools, which is a modeling tool for colored and timed PNs.

***Keywords*—**Cloud computing, elasticity, elasticity controller, petri nets, scaling in, scaling out.

I. Introduction

CLOUD computing is an important area of research that is the subject of several scientific contributions. It is a new model based on a 'use on demand' and 'pay as you go' principle. Cloud systems are a set of services provided by cloud providers. These provide virtualized resources (servers, virtual machines, services, etc.) as on-demand services. Cloud services are spread over three levels which are; Infrastructure as a Service (IaaS), Platform as a service (PaaS) and Software as a Service (SaaS). These services use cloud components (such as databases, containers, VMs) which themselves use cloud resources (such as CPU, memory, network) [1].

The main roles and objectives of cloud providers are the supply of resources to meet customer demands, meet the service level constraints SLA (Service Level Agreement), guarantee a good quality of service and at the same time minimize downtime and energy consumption [2], [3].

The variation in the demand for services offered by the cloud leads to a variation in the resources provided by the addition or removal of the available resources, hence the need for an elastic system.

Fig. 1 illustrates cloud elasticity; auto scaling is the ability ofa system to quickly adapt the resources provided to the resources requested so as not to violate SLA constraints and the cloud service continues to run smoothly even when workload scales up or down, avoiding underuse and overuse of resources [4].

To properly manage elastic cloud systems, the supply of resources must be controlled by a resource manager also called an elasticity controller. The latter allows system sizing and the automatic implementation of action plans to meet the real demands of the new system state [6].

IBM [7] defines Autonomic Computing as the ability to manage computing resources that automatically and dynamically respond to the requirements of the business based on SLA. Autonomic management is usually presented as a Monitor, Analyze, Plan and Execute (MAPE) loop as shown in Figs. 2 and 3. The different functions of the autonomic control loop are defined as:

- M: Monitor function that provides the mechanisms to collect, aggregate, filter and report monitoring data collected from a managed resource.
- A: Analyze function that provides the mechanisms that correlate and model complex situations and allow the autonomic manager to interpret the environment and the current state of the system, and predict future situations.
- P: Plan function that provides the mechanisms that construct a plan of actions needed to achieve a certain goal, usually according to some guiding strategies.
- E: Execute function that provides the mechanisms to control the execution of the plan over the managed resources.

In this article, we propose a formal model for modeling the static and dynamic aspects of a cloud system and its elasticity controller. We used PNs to model the structural and behavioral aspects of a cloud system and its elasticity.

The places model the nodes and all components of the cloud and the transitions are the rules to be applied so that the system passes from one state to another. Additionally, we add to the CPN the notion of time which is important to model the mechanisms of elasticity. The parameter of time is important to define the availability of VMs, the variation in the workload, the capacity of the servers, and the adequate time for carrying out scaling.

W. Louhichi is with High Institute of Technological Studies, Department of Computer Sciences, Gabes Tunisia (phone: 00216-29 59 05 26; e-mail: louhichi_walid2@yahoo.fr)

M. Berrima is assistant professor with the Department of Computer Science, Faculty of Sciences of Monastir, Sousse University, Tunisia (e-mail: berrima.mouheb@gmail.com).

N. Benrajeb is a professor with the Computer Sciences Department, University of Carthage, Tunisia (e-mail: narjes.benrajeb@gmail.com).

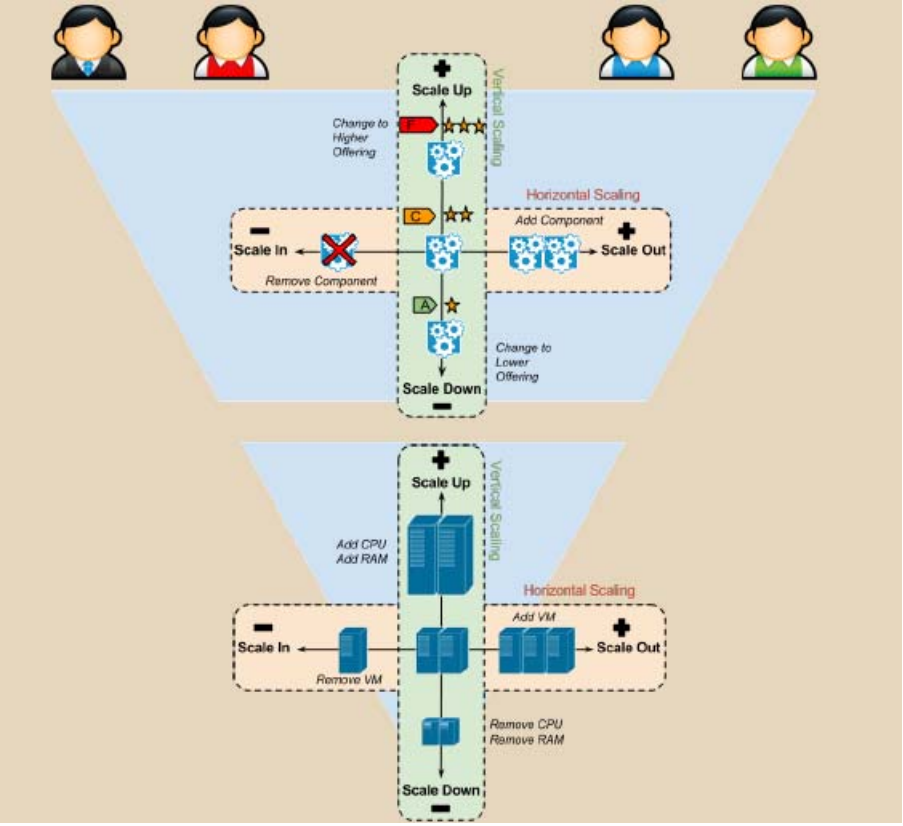

Fig. 1 Auto-Scaling [5]

The rest of this paper is planned as follows. Section II presents the associated work of the formal modeling of the elasticity of cloud computing. In Section III, we introduce a formal modeling of cloud computing with timed and colored PNs. In Section IV, we define cloud system elasticity and specially the horizontal scaling. We define and model the mechanism of scaling in and scaling out elasticity with colored and timed PNs in Section V. The established model is edited, verified and simulated by an example of configuration with the CPN/Tools in Section VI. In Section VII, we present CPN/Tools. Finally, Section VIII summarizes and reports our reviews in future works.

Fig. 2 Autonomic Control loop for cloud resource [8]

II. Related Work

With the expanding importance in cloud computing, several research works related to cloud computing and resources management have been proposed. One of the most successful fields of research in cloud computing is elasticity. Many studies in the literature have addressed the formal modeling of elasticity in cloud computing. These works are divided into two parts; the first concerns the methods or formalisms of modeling while the second is focused on the validation, verification and simulation of the model generated by the first part.

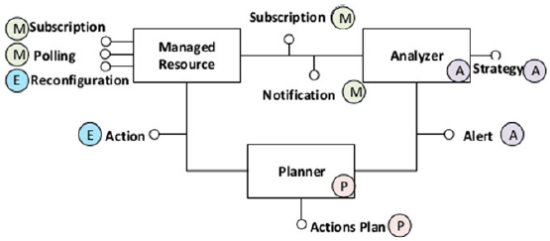

Fig. 3 Autonomic infrastructure for cloud resource [8]

In [9], a formalism of Bigraphs Reactive System (BRS) is simulated with Graph Transformation System (GTS). Reactive systems are based on the definition of sets of rules that decide the behavior of systems. Brs is the formalism for describing and analyzing mobile computation and pervasive systems [10]. Moreover, in [11], the authors proposed a Brs approach for modeling an elastic cloud system. Bigraphs represent the static aspects of a cloud system and bigraphical reactions rules for a description of the behavior of system according to the cases. Proposed model is illustrated and confirmed by a case study. The

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

same authors in [12] have presented a formal modeling and verification approach for cloud systems elasticity based on bigraphical reactive systems and Maude language. Bigraph CS represents the structure of the cloud and Maude language is used for describing and simulating the reactions rules.

Others research initiatives [13] use CPNs to demonstrate load balanced scheduling and model reliability in network transaction processing. The load balancer is the essential component which makes it possible to adapt the resources provided to the workloads. The proposed model with CPN present the distribution and execution of computing environments like cloud computing. The authors consider that the combination of PNs with programming languages provides a powerful modeling technique. Additionally, [14] introduced a Dynamic Scalable Stochastic Petri Net (DSSPN) model that studies the performance of cloud systems. This model is validated by a series of experiments through Stochastic Petri Net Package (SPNP). Cloud computing system performance assessment examines the liaison between system structure, workload, and performance index. Authors in [15] have presented a performance model for evaluation of cloud infrastructure performance based on stochastic PNs. This performance study concerns services deployed in virtual machines and containers in cloud infrastructures. In [16], Narayanan and Cherukuri designed an extension of information integration architecture with a Cloud Data Lake and verified it with Petri Net. This approach allows a vertical scaling in real-time data and to conduct an adequate way of using data resources. Authors in [17] proposed a model of elasticity controller for the automatic management of cloud infrastructure. This pattern is elaborated by CPNs. The performance study of this elasticity controller is made on three different workloads and compared with other approaches. This appraisement shows that the designed model ameliorates the use of resource, elasticity and response time.

III. Model Cloud System with Timed and Colored Petri Nets(TCPN)

A. Definition of TCPN

PNs associate characterizing function and formal verification methods with pervasive possibilities for quantitative treatment. They are a graphical mathematical modeling tool application to many systems [18]. They are favorable tools for the modeling of systems according to their synchronization, centralization, dispersion, parallelism and determination. A PN is identified as a particular kind of bipartite directed graph populated by three types of objects. They are places, transitions, and directed arcs connecting places and transitions (see Fig. 4).

A temporized PNs is a CPN with time parameters. TCPN is a tuple TCPN = (CPN, D,d_0); where, D is the set of time values and $d_0$is an element of D called the departure date [19].

TCPNs [20]:

$$A\ TCPN\ is\ a\ 10-tuple, TCPN\ = \{P, T, A, O, \sum, N, C, G, E, I\}$$

where

- $P = \{p1, p2, \dots, pm\} is\ a\ finite\ set\ of\ places, where\ m > 0;$
- $T = \{t1, t2, \dots, tn\}\ is\ a\ finite\ set\ of\ transitions\ with\ P \cup T \neq \emptyset\ and\ P \cap T = \emptyset, where\ n > 0;$
- $A: P \times T \rightarrow$ $N\ is\ an\ input\ function\ that\ defines\ a\ set\ of\ directed\ arcs\ from\ P\ to\ T, where\ N = \{0, 1, 2 \dots\};$
- $O: T \times P \rightarrow N$ is an output function that defines a set of directed arcs $from\ T\ to\ P. M0: P \rightarrow N$ is the initial marking.
- $\sum is\ a\ finite\ set\ of\ non-empty\ color\ set$
- N a node function defined from A into $P \times T \cup T \times P$
- C:$P \rightarrow$ $\sum$is a color set function that assigns a color set to each place
- $G: T \rightarrow EXPR\ is\ a\ guard\ function\ that\ assigns\ a$ guard to each transition t such that:$[Type(G(t)) = Bool \wedge Type\left(Var(G(t))\right) \subseteq \sum]$
- $E: A \rightarrow EXPR\ is\ an\ arc\ expression\ function\ that\ assigns\ an\ arc\ expression\ to\ each\ arc\ a\ such\ that: \forall a \in A: \left[Type(E(a)) = C(p(a))MS \wedge Type\left(var(E(a))\right) \subseteq \sum\right]\ where\ p(a) is\ the\ place\ of\ N(a).$
- $I: P \rightarrow EXPR\ is\ an\ initialization\ function\ that\ assigns\ an\ initialization\ expression to\ each\ place\ p\ such\ that:$ $\forall p \in P: \left[Type(I(p)) = C(p)MS \wedge Type(Var\ I(p))\right) \subseteq \emptyset]$

A network marked N is a couple (R, M) made up of a Petri R network and a defined tagging application on P and with values in N (i.e., marking the network at an instant given).

A transition t is pull-able (or passable, or validated) when:

$$\forall p \in \Gamma^{-1}(t) M(p) \geq pre(p, t)$$

Let N= (R, M) be a Petri Net marked with T transitions and P places. P. The crossing of a transition t of T validated in the M marking leads to the M1 marking:

$$\forall p \in P, \forall t \in T, M_1(p) = M(p) + C(p, t)$$
$$\forall p \in P, \forall t \in T, M_1(p) = M(p) + Post(p, t) - Pre(p, t)$$
$$M1\ (p)\ =\ M\ (p)\ +\ Post\ (p, t)\ -Pre\ (p, t)$$
$$We\ then\ denote\ M\ [t > M1.$$

B. Petri Net Model of Cloud Systems

Our paper is inspired from [13] and [11] which deal with the modeling and verification of cloud systems based on Bigraphical reactive systems. Bigraphs are associated with reaction rules to form bigraphical reactive systems (BRS) that can be applied to rewrite bigraphs. PNs are a bipartite graph and the two families of vertices are the places and transitions. Table I presents a correspondence between the concepts of BRS, Cloud and PNs.

PNs are graphical and mathematical tools for modeling and verifying the dynamic behavior of different types of systems like networks, communications and industrials systems. In this paper, we model a cloud system using a PN model. The proposed graph shows the different physical and logical components of a cloud system and their interaction.

In our simplified model (Fig. 4) for the representation of a cloud system with the Petri network, we choose the following composition:

- One Data center.
- Two servers hosted in a Data center.

- Two virtual machines can be hosted in a server.
- The applications can be deployed in virtual machines by users.

C. Color Sets

The color sets used for the elaboration of the PNs model for the verification and modeling of the elasticity of cloud computing are as follows:

- Colset VMIN= with A|B timed;
- Closet SEIN = with SR1|SR2 timed;
- Colset VMOUT = product VMIN* INT*INT timed;
- Colset SEOUT = product SEIN * INT timed;
- Closet VMSE = product VMIN * SEIN * INT * INT * INT timed;
- Closet Etat= product SEIN* VMIN* INT timed;
- Var R,r,t : int;

TABLE I
CORRESPONDENCE BETWEEN BRS CONCEPTS, CLOUD AND PNS

Cloud element	Bigraphical concepts	Petri Net
	Cloud system structure	
Cloud system	Bigraphs CS	$R = \{P, T, Pre, Post\}$
Client, data center, load balancer, server, container, VM, service	Nodes	Places: $p \in P$
Interaction	Edges	Transitions: $t \in T$
	Cloud system elastic behavior	
Elasticity action	Reaction rule: $Cs \xrightarrow{R} CS'/R = (R, R', n)$	$\forall p \in \Gamma^{-1}(t)M(p) \geq pre(p,t).$ $M0[\sigma > M1$
	Representation	
	Bigraph	Bipartite graph

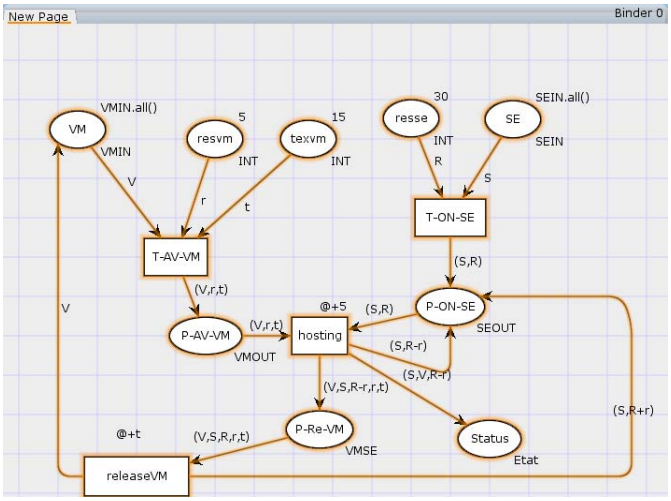

Fig. 4 Petri Net Cloud Model

TABLE II
THE LIST OF PLACES

Place	Designation
VM	The place that contains all virtual machines.
resvm	The place that contains the amount of resources consumed by each virtual machine.
texvm	The state that indicates the execution time of each virtual machine.
SE	The place that contains all servers.
resse	The place that indicates the capacity of each server.
P1	The state that shows all virtual machine ready for hostingin available servers.
P2	The place that shows active servers.
P3	The place that contains all hosted virtual machines, the hosting server and their capacities.
P4	The place that contains the chronological state of virtual machines hosting in the servers.

TABLE III
THE LIST OF TRANSITIONS

Transition	Designation
T1	This transition gives each virtual machine its execution time and the amount of resources required.
T2	This transition allows starting a server.
T3	This transition hosts a virtual machine in a server. The crossing of this transition lasts five units of time.
releaseVM	This transition removes a virtual machine from a server. The crossing of this transition updates the capacity of server after checking the end of virtual machine execution time.

All these color sets are timed.

- The color VMIN defines the different types of virtual machines used and hosted in the servers of the data center.
- The timed color set SEIN defines the different types of servers in cloud datacenters.
- The color set VMOUT is used to present for each virtual machine the quantity of resources consumed (r) by its hosting and its duration of execution (t).
- SEOUT, is a color set that indicates the capacity of each server in the data center.
- Color set VMSE is a multi-set that contains five fields. The first field is marked with V is of type VMIN and represents the hosted virtual machine. The second field is denoted with S is of type SEIN and indicates the hosting server of virtual machine. The third field is declared R (R=R-r) and it represents the new sever capacity after hosting the virtual machine V.The fourth field is for the representation of virtual machine capacity and is defined by the variable r. Finally, the last field is denoted t for execution time of the virtual machine.
- Color set Etat is used as a server status dashboard and it contains three fields. The first one is denoted with S is of

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

type of server. The second field is declared V and represents the type of virtual machine hosted in server S. The last field is denoted R-r and represents the new capacity of server after hosting the virtual machine V.

The characters @ and @+ operators are used to add time stamps to the tokens.

- x@t: attach the time stamp t to the token x.
- Example X@5: the token X is available in a place at the time stamp 5.
- m@+i: adds a delay i to each token of multi-set m and returns a timed multi-set.
- If we put in a transition @+5 i.e., crossing this transition lasts 5 time stamp units.

IV. Cloud System Elasticity

A. Definition of Cloud Elasticity

Remember that the elasticity is the variation of resources to the workload. In order to adjust the supply of resources in proportion to demand, elasticity controller must be used. It makes it possible to set up a set of rules allowing the resizing of cloud resources to meet demand with minimum costs and energy consumption. Scaling is a strategy adopted by considering several variables such as the actual workload and the resources allocated and available, and even the prediction of the future state of the cloud system [21].

The elasticity controller is an infinity cycle for cloud system control which has sensors for raising awareness of the system state and effectors for performing the reaction necessary for the actual situation. The elasticity controller plug is called MAPE and it is that of IBM's autonomic control loop.

The elasticity controller operation is defined by a PN model which represents the dynamicity of an elastic cloud. In a PN, the places of entry for a transition represent the preconditions, input data, resources required or input buffer. The transition is the event, treatment, job, activity or process. The exit places post conditions such as output data, resources released or output buffer.

B. Horizontal Scaling

The horizontal scaling is ensured by two mechanisms;the first, scaling out, is the addition of resources such as VM if additional resources are needed (in case of under-supply), and the second, scaling in, is the removal of a resource in the event of over-supply. Another method of ensuring the elasticity of cloud systems is migration. This operation is the change of the hosting of a virtual machine from one physical machine to another.

Horizontal Scaling Algorithm

```
Begin
        Sizing (System, Request, offer)
        If (over supply)
                Then
         Scaling IN;
                Elsif (under supply)
         Scaling OUT;
                Else
                NULL;
        End if;
End
```

V. Modeling Horizontal Scaling

A. Scaling OUT

The under sizing is due to an undersupply of resources by the cloud provider and affects the quality of service (QoS) and even compliance with the SLA level constraint. Under-supply is the case where the amount of resource allocated is very low compared to that requested. In this case, the solution is the duplication of the instances of the resources without reconfiguration; this operation is called Scaling Out.

TABLE IV
List of Places for Scaling OUT

Place	Designation
VM	The place that contains all virtual machines (A,B).
resvm	The place that indicates the amount of resource consumed by the execution of each virtual machine.
SE	The place that contains all servers.
resse	The place that indicates the capacity of each server.
P1	The place which contains the list of available virtual machines and the quantities of resources necessary for their execution.
P2	The place that contains the list of running servers and the amount of remaining resources.
P4	The place which chronologically indicates the hosting of virtual machines in the different servers and their remaining quantities of resources.

TABLE V
List of Transitions for Scaling OUT

Place	Designation
T1	The transition which allocates to each virtual machine the quantity of resource necessary for its execution.
T2	The transition which attributes to each server its capacity in terms of resource quantity.
T3	The transition which ensures the hosting of a virtual machine in a server. The execution of this operation takes five units of time. This transition is passable only when the quantity of resources available in the destination server is greater than or equal to the quantity of resources consumed by the machine in question.

Definition. Let S=$\langle N|M\rangle$ be a net system at t0 and let$p \in P$, the duplication of p in S by a new placep^c , noted as D(S, p, p^c), is a new net system S'=$\langle N'|M'\rangle$ at t1.

T0: beginning time.

T1: end of duplication time.

- $P' = P \cup \{p^c\}$
- $T' = T \cup T''\ with\ T'' = \{t^c | t \in (*_{p \cup p^*})\}$
- $Pre' : P' \times T' \to \{0,1\}$
- $Post' : T' \times P' \to \{0,1\}$
- $M' : P' \to N\ with\ M'(p') = M(p') if\ p' \neq p^c$ and 0 otherwise.
- $pre'(p', t') =$
$$\begin{cases} pre(p', t') p' \in P\ and\ t' \in T \\ pre(p', t)\ t \in T\ and\ t' \in (T' \backslash T)\ and\ p' \in (P \backslash \{p\}) \\ pre\ (p, t)\ t \in T\ and\ t' \in (T' \backslash T)\ and\ p' = p^c \\ 0\ otherwise \end{cases}$$
- Post′(t′, p′) =

$$\begin{cases} Post(t',p')p' \in P \; and \; t' \in T \\ Post\,(t,p')\; t \in T \; and \; t' \in (T'\backslash T) \; and \; p' \in (P\backslash\{p\}) \\ Post\,(t,p)\; t \in T \; and \; t' \in (T'\backslash T) \; and \; p' = p^c \\ 0 \; otherwise \end{cases}$$

B. Scaling IN

Over sizing is due to an oversupply of resources by the cloud provider. We speak of an over-sizing or over-supply when the quantity of resources offered is much greater than the demand. This difference is due to a rapid variation in the demand for resources. The excess of available resources leads to a waste of energy which will have an impact on the cost of energy consumption.The solution to this problem is to remove some resources to balance the resources against the workload; this operation is called Scaling IN.

Definition. Let S = $\langle N, M\rangle$ be a net system at t0 and let p,p^c be two places in N with $p \neq p^c$, the consolidation of p^c in p, noted as $C(S,p,p^c)$, is a new net system $S' = \langle N', M'\rangle$st at t1.

T0: beginning time.

T1: end of removing time.

- N': is the net N after removing the placep^c.
- M':$p' \rightarrow Nwith\; M'(p) = M(p) + M(p^c) and M'(p') = M(p') if p' \neq p.$

TABLE VI
LIST OF PLACES FOR SCALING IN

Place	Designation
VM	The place that contains all virtual machines (A,B).
resvm	The place that indicates the amount of resource consumed by the execution of each virtual machine.
texvm	The place which indicates the duration of execution of a virtual machine.
SE	The place that contains all servers.
resse	The place that indicates the capacity of each server.
P1	The place which contains the list of available virtual machines and the quantities of resources necessary for their execution.
P2	The place that contains the list of running servers and the amount of remaining resources.
P3	This place contains the list of running virtual machines.
P4	The place which chronologically indicates the hosting of virtual machines in the different servers and their remaining quantities of resources.

TABLE VII
LIST OF TRANSITIONS FOR SCALING IN

Place	Designation
T1	The transition which allocates to each virtual machine the quantity of resource necessary and the duration for its execution.
T2	The transition which attributes to each server its capacity in terms of resource quantity.
T3	The transition which ensures the hosting of a virtual machine in a server. The execution of this operation takes five units of time and it is not passable only when the quantity of resources available in the destination server is greater than or equal to the quantity of resources consumed by the machine in question.
releaseVM	This transition allows the release of a virtual machine following the end of its execution time.

VI. SIMULATION OF SCALING IN AND SCALING OUT

A. Scaling OUT

The initial marking of this PN is as followsand shown in Fig. 6:

- Two tokens in the VM place representing a type A of virtual machine. The two tokens A in place VM are available from time t = 2.
- Two tokens in the resvm place representing the amount of resource consumed by hosting a virtual machine (5 units).
- Two tokens (SR1, SR2) in the SE place representing the different servers existing in the data center.
- One token in the resse place representing the capacity of server (30).

In Fig. 6, the execution of T1, T2 then T3 from initial marking makes it possible to host virtual machine A in server SR1.The duration of the crossing of T3 is 5 units of time. Crossing the transition produces a token in P4 which indicates the hosting of the virtual machine A in the server SR1 at the instant t = 7 and the updating of the quantity of resources available on the server(R=R-r= 25): R: capacity of server (30 in the initial marking); r: the quantity of resources consumed by the hosting of virtual machine A.In the same way, this crossing produces a token in P3 representing the new capacity of the server SR1 after hosting the virtual machine A (R=R-r=25) at the instant t = 7.We notice that T3 consumes a single token A and remains another available token (1' (A,5) @2.

Crossing the transition T3 a second time, Fig. 7, allows the duplication of the virtual machine A on the server SR1 at the instant t = 12.Crossing the transition produces a new token in P4 which indicates the hosting of the virtual machine A in the server SR1 at the instant t = 12 and the updating of the quantity of resources available on the server(R=R-r=20).

P3 is a place for the history of host machines in the servers.In the same way, this crossing produces a token in P3 representing the new capacity of the server SR1 after hosting the virtual machine A (R=R-r=20) at the instant t = 12.

B. Scaling IN

The initial marking of this PN is as follows and shown in Fig. 8:

- Two tokens in the VM place, representing type A and type B of a virtual machine, are available from time t = 2.
- One token in the resvm place representing the amount of resource consumed by hosting a virtual machine (5 units).
- One token in texvm place representing the duration of execution of one virtual machine.
- Two tokens in (SR1, SR2) in the SE place representing the different servers existing in the data center.
- One token in the resse place representing the capacity of each server (30).

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

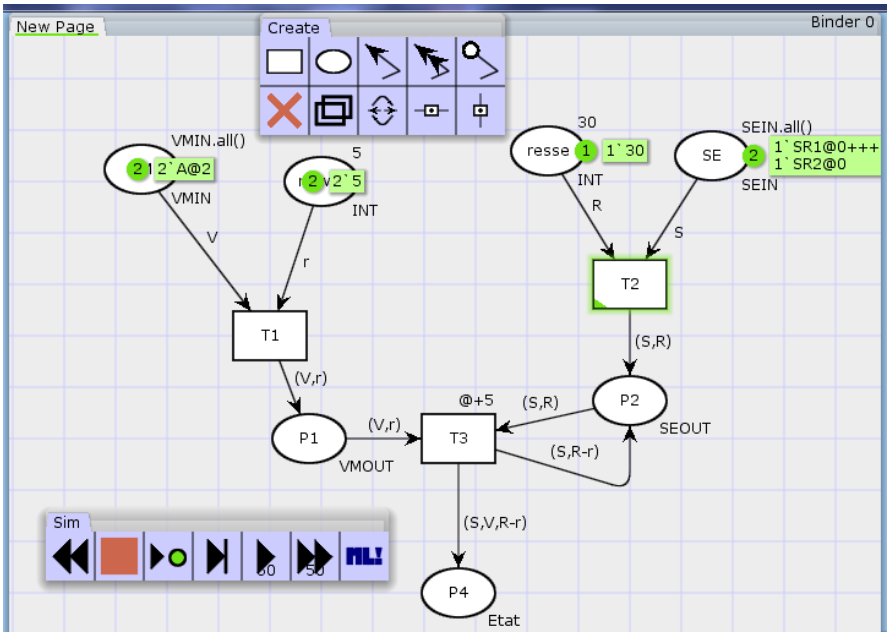

Fig. 5 Initial marking of TdCPN model for Scaling OUT

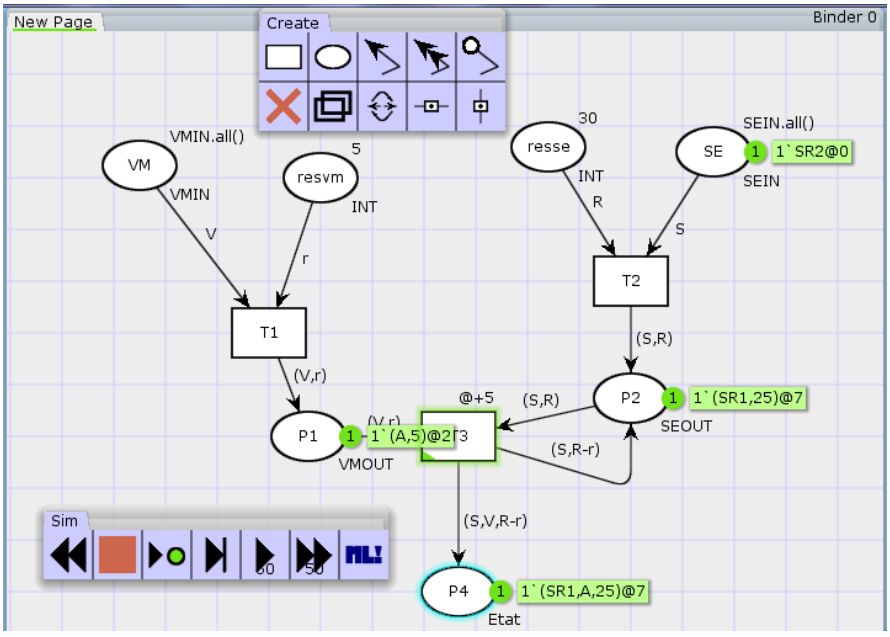

Fig. 6 Hosting a VM A into Server SR1

Crossing T1 and T2 gives a token in P1 place for a VM ready to run (duration of running =15) and a token in P2 place for running server (see Fig. 9).

The duration of the crossing of T3 is 5 units of time. Crossing the transition T3 produces a token in P4 which indicates the hosting of the virtual machine A in the server SR2 at the instant t = 5 and the updating of the quantity of resources available on the server (R=R-r=25) (see Fig. 10).

P2 is the place that contains the running servers and his capacity. In the same way, this crossing produces a token in P3 representing the new capacity of the server SR1 after hosting the virtual machine A (R=R-r=25) at the instant t = 5.

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

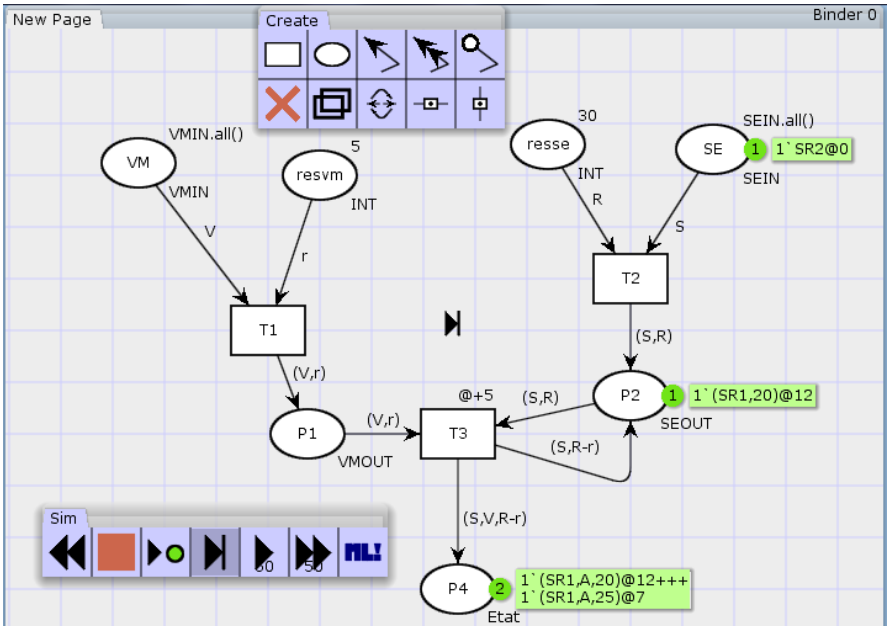

Fig. 7 Scaling OUT: Duplication of VM into a server

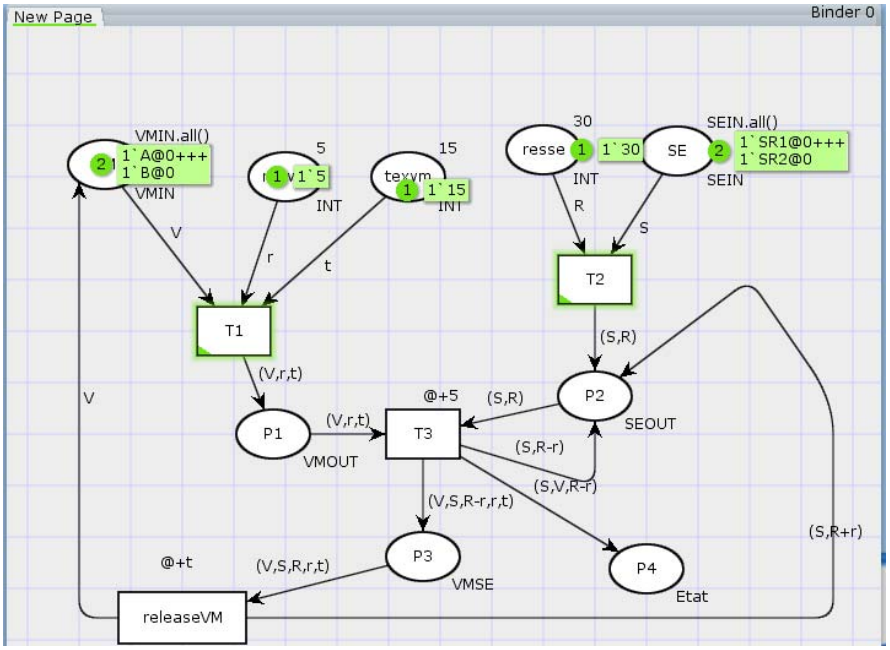

Fig. 8 Initial marking of TdPN model for Scaling IN

The P3 place contains the list of running virtual machines. R: capacity of server, r: the quantity of resources consumed by the hosting of virtual machine A, t: the duration of execution of one virtual machine.

In Fig. 11, the transition release VM can only be crossed when the virtual machine has finished its execution time. Crossing the transition release VM frees the virtual machine A at time t = 20 modeled by a token in the VM place and updates the server capacity SR2 (R=R+r).

VII. Support Tools

The model presented in this work is edited by CPN/Tools. CPN/Tools were originally developed by the CPN Group at Aarhus University from 2000 to 2010 [22]. CPN/Tools comprise two main components, a graphical editor and a backend simulator component. CPN/Tool is a tool for editing, simulating, and analyzing basic PNs, timed PNs and colored PNs.

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

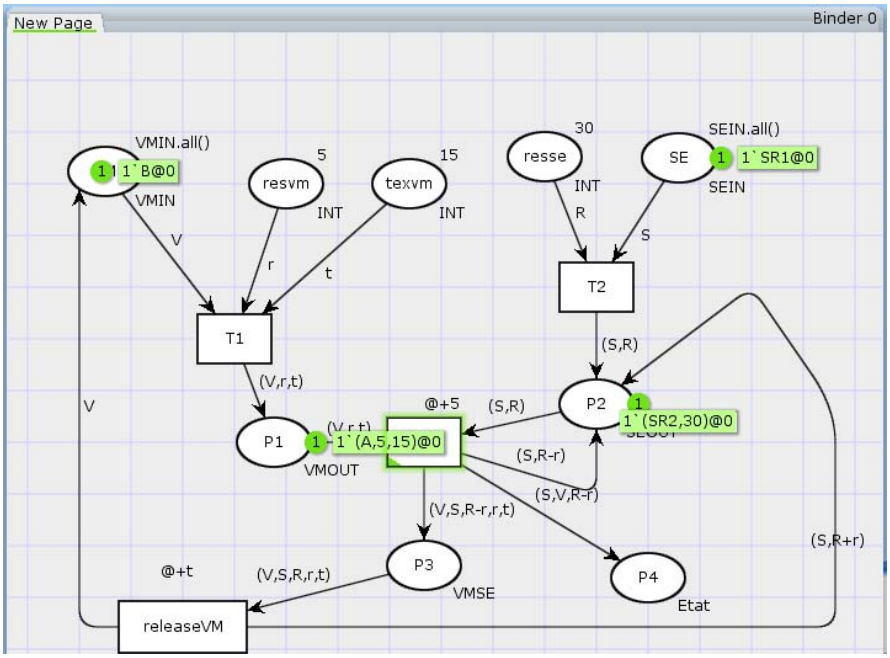

Fig. 9 Available VM A for Hosting in SR2

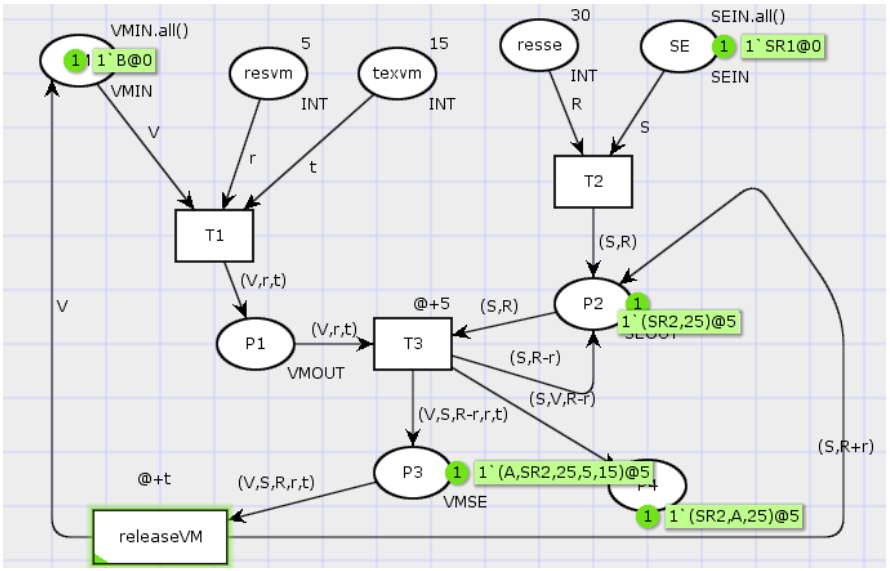

Fig. 10 Hosting a VM A into Server SR2

VIII. Conclusion and Future Work

In this paper, we have presented two formal models for the modeling of horizontal elasticity in cloud systems. The established models are based on the colored and timed PNs. The PN is a bipartite graph of which we particularize the two families of vertices: places and transitions. PNs are a modeling language that represents the structural and behavioral aspects of complex systems. The places are the preconditions of transitions and any crossing gives a new marking or a new state of the system. We used CPN Tools for building models. CPN Tools is a graphical editor and simulator of colored and temporized PNs.

The proposed approach is validated by the simulation of two examples, one for scaling in and the other for scaling out. The scaling in example shows how to remove a virtual machine from a server after the end of the execution cycle and the second model shows how to duplicate a virtual machine in a server. In the studied cases, we clearly show the time and the updating of the server capacity.

In future work, we aim to extend our work by an integrated solution for auto-scaling, vertical and horizontal scaling. In addition, the aim is to simulate and evaluate the proposed approach in a real environment of cloud system.

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf

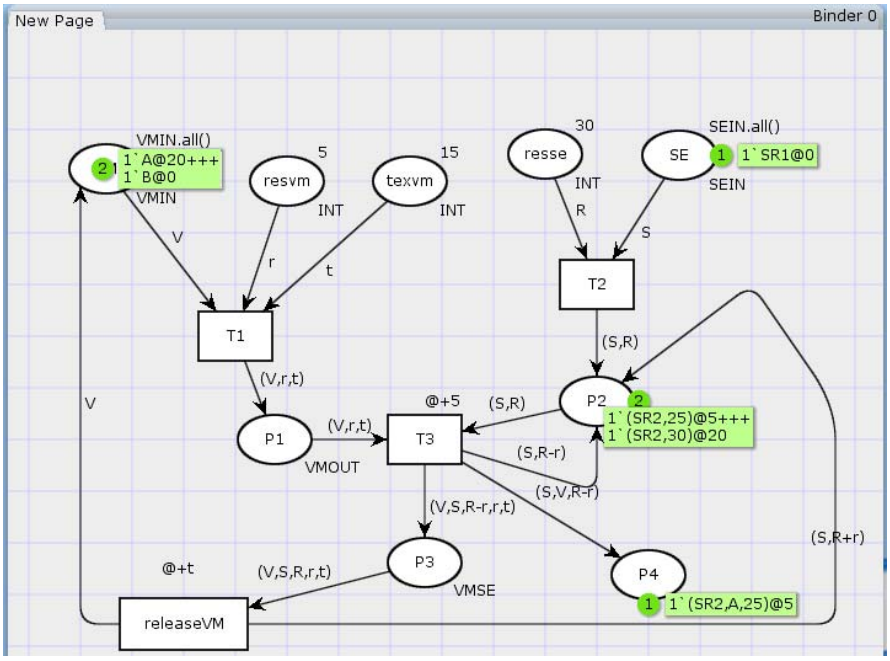

Fig. 11 Release VM A from server SR2: Scaling IN

REFERENCES

[1] M. Amziani, K. Klai, T. Melliti, and S. Tata,"Time-based Evaluation of Service-based Business Process Elasticity in the cloud,"IEEE 5th International Conference on Cloud Computing Technology and Science,2013.

[2] S. Dustdar, Y. Guo, B. Satzger, and H. Truong, "Principles of elastic processes. IEEE Internet Computing,"vol. 15, 2011, pp. 66–71.

[3] J. Geelan, M. Klems, R. Cohen, J. Kaplan, D. Gourlay, P. Gaw, D. Edwards, B. de Haaff, B. Kepes, K. Sheynkman, O. Sultan, K. Hartig, J. Pritzker, T. Doerksen, T. von Eicken, P. Wallis, M. Sheehan, D. Dodge, A. Ricadela, B. Martin, B. Kepes, and I. W. Berger,"Twenty-One Experts Define Cloud Computing,", 2009.

[4] F. Zhang, X. Tang, X. Li, S.U. Khan, and Z. Li,"Quantifying cloud elasticity with container-based auto scaling. Future Generation Computer Systems,"vol. 98, 2019,pp. 672–681.

[5] S. Dupont, J.Lejeune, F. Oliveira Jr. and T. Ledoux,"Experimental Analysis on Autonomic Strategies for Cloud Elasticity," In IEEE International Conference on Cloud and Autonomic Computing (ICCAC), Cambridge, Massachusetts, USA, pp. 21-25, September 2015.

[6] N. Herbst, S.Kounev, and R.Reussner,"Elasticity in cloud computing: what it is, and what it is not,"in Proceedings of the 10th International Conference on Autonomic Computing, USENIX, San Jose, 2013, pp. 24-27.

[7] B. Jacob, R. Lanyon-Hogg, D.K. Nadgir, and A.F. Yassin, "A Practical Guide to the IBM Autonomic Computing Toolkit," IBM Redbooks, http://www.redbooks.ibm.com/, 2004.

[8] M. Mohameda, M. Amziania, D. Belaïd, S. Tata, and T. Melliti,"An autonomic approach to manage elasticity of business processes in the Cloud," Future Generation Computer Systems, vol. 50, 2015,pp. 49–61.

[9] A.Gassara, I.B. Rodriguez, M. Jmaiel, K.Drira,"Executing bigraphical reactive systems", journal of Discrete Applied Mathematics, 2018.

[10] R. Milner, "The Space and Motion of Communicating Agents", Cambridge University Press, 2009, pp. 1–191.

[11] K. Khebbeb, H. Sahli, N. Hameurlain and F. Belala, "A BRS Based Approach for Modeling Elastic Cloud Systems", Springer International Publishing AG, part of Springer Nature 2018

[12] L. Braubach et al. (Eds.): ICSOC Workshops 2017, LNCS vol. 10797, pp. 5–17.

[13] H. Sahli, F.Belala and C.Bouanaka,"A BRS-Based Approach to Model and Verify Cloud Systems Elasticity", Procedia Computer Science, vol. 68, 2015,pp. 29 – 41.

[14] D. P. Mahato and R.S. Singh, "Load balanced scheduling and reliability modeling of grid transaction processing system using colored Petri nets",ISA Transactions, vol. 84, January 2019, pp. 225-236.

[15] H. He, S. Pang and Z. Zhao, "Dynamic Scalable Stochastic Petri Net: A Novel Model for Designing and Analysis of Resource Scheduling in Cloud Computing". Hindawi Publishing Corporation Scientific Programming, vol. 2016(3), January 2016, pp. 1-13;

[16] R.R Yadav, G.A.S. Campos, E.T.G. Sousa, F.A. A. Lins, "A Strategy for Performance Evaluation and Modeling of Cloud Computing Services", vol. 26, num.1 (2019), pp.78-90.

[17] M. Narayanan, A.K. Cherukuri," Verification of Cloud Based Information Integration Architecture using Colored Petri Nets", I. J. Computer Network and Information Security, 2018, 2, pp. 1-11.

[18] T. Murata, "Petri Nets: Properties, Analysis and Applications," Proceedings of the IEEE, vol. 77, no. 4, 1989, pp. 541–580.

[19] K. Jensen, Coloured Petri Nets – Basic Concepts, Analysis Methods andPractical Use, vol. 2, Springer-Verlag, Berlin, 1995, p 174.

[20] Jensen, K.; Kristensen, L.M. Formal Definition of Timed Coloured Petri Nets. In Coloured Petri Nets; Springer:Berlin/Heidelberg, Germany, 2009; pp. 257–271.

[21] J.S. Lee, and P-L. Hsu, "Implementation of a remote hierarchical supervision system using Petri Nets and agent technology," IEEE Transactions of Systems, MAN, and Cybernetics – Part C: Applications and Reviews, vol. 37, no. 1, 2007, pp. 77–85.

[22] M. Beaudouin-Lafon, W.E. Mackay, P. Andersen, P. Janecek, M. Jensen, M. Lassen, K. Lund, K. Mortensen, S. Munck, A. Ratzer, K. Ravn, S. Christensen and K. Jensen", CPN/Tools: A Post-WIMP Interface for Editing and Simulating Colored Petri Nets", Proceeding of 22nd International Conference on Applications and Theory of Petri Nets, 2001(ICATPN 2001), LNCS vol. 2075, Springer, 2001.pp 71-80.

Open Science Index, Electrical and Computer Engineering Vol:16, No:3, 2022 publications.waset.org/10012425.pdf