

Modeling and Analysis of Enterprise Cloud Bus Using a Petri Net Based Approach

Gitosree Khan, Sabnam Sengupta and Anirban Sarkar

Abstract Agent based Cloud computing technology has become a predominant domain nowadays due to rapid increase in enterprise service applications that are built from various services offered by different cloud service providers. In our previous work, we have proposed an abstraction layer of SaaS architecture for Inter-cloud environment, called Enterprise Cloud Bus System (ECBS) and modeled it using Unified Modeling Language (UML). In this work, we have proposed a conceptual architecture of Multi-agent based Enterprise Cloud Bus System (ECBS) and modelled its dynamics using Petri Net (ECBP) based approach. The proposed approach will be beneficial for analyzing low level Petri Nets of any cloud based models and has the advantage of being able to model dynamic facets of the Multi-agent based cloud system that can handle multiple agent operations and its execution logics. Further, the proposed ECBP model is capable to analyze the behavioral facets of Enterprise Cloud Bus System like, fairness, boundedness, liveness, safeness, etc. in a dead lock—free environment.

Keywords Cloud computing • Enterprise Cloud Bus System • Behavioral analysis • Enterprise Cloud Bus Petri Net • PIPE Petri Net tool

G. Khan (✉) • S. Sengupta

B. P. Poddar Institute of Management & Technology, Kolkata, India
e-mail: khan.gitosree@gmail.com

S. Sengupta
e-mail: sabnam_sg@yahoo.com

A. Sarkar
National Institute of Technology, Durgapur, India
e-mail: sarkar.anirban@gmail.com

1 Introduction

Cloud computing technology delivers on demand service over the network as per user's requirement. In recent days, due to rapid increase of cloud and its services as well as the growing interest of enterprise towards service oriented computing [1–4] discovery and selection leads to the problem of flexible and scalable access to computing resources. Lots of researchers [5, 6] discuss over the architectural design of cloud computing and its applications. Many of them focus on the vision, challenges and architectural elements [7] of Inter-cloud environments. The work in [8, 9] covered on dynamic provisioning, deployment and adaptation of dynamic Multi-cloud behaviour systems. The author in [10] proposes a new approach for dynamic autonomous resource management in cloud computing.

In recent days, challenges and research directions of agent oriented software engineering has been explained in [11]. Formalization and analysis of Multi-agent based software system has been proposed in [12]. There are few works [13–15] based on measurement and validation of complexity metrics of Multi-agent based system architecture. Further the author in [16] models the Multi-agent system dynamics using graph semantic based approach. Moreover, modeling and design of agent based hybrid and flexible Multi-cloud architecture [17, 18] has emerged as one of the most challenging domains in cloud computing. Thus, as our previous work are based upon such MAS based Multi-cloud architecture. In [19], we have proposed a Multi-Agent based abstraction layered architecture, called Enterprise Cloud Bus (ECB) and modeled its structural components using UML 2.0. [20], to make the cloud based system more reliable and robust. Few of our earlier work are based on service registration and discovery mechanism in ECBS [21–23] which helps to identify services during run time. Further, in [24, 25] scheduling of services and its performance analysis in ECBS has been proposed. However, UML cannot be used for automatic analyses and simulation of Inter-cloud architecture, because of its Semi-formal nature.

Since, the UML modeling lacks to exhibit the dynamism of internal behavior of the system. PIPE [26, 27] is a Platform Independent Petri Net Editor tool that helps to analyze low level Petri Nets and has the advantage of being able to model dynamic behavior of the Multi-agent based cloud system and handle Multiple agent operations and its execution logics. In [28–32] the author proposed a Petri net based approach for modeling and analysis of agent oriented system. In such a domain, few of the research work [33] are done on modeling of Inter-cloud architecture using Petri Net tool.

This work enhances the modeling and analysis of Multi-cloud architecture (*ECBS*) using Petri Net based approach (*ECBP*), in order to address the issues of the dynamic facets of the system. The proposed method helps to analyze and verified the behavioral properties of the system like, reachability, safeness, boundedness, liveness, etc. This kind of formal approach is helpful for effective design of *MAS* based Inter-cloud architecture by providing maximum utilization of resources on demand through scheduling agent, automation of resources and dynamic provisioning of all types of heterogeneous cloud services. Further, a case study has been explained based on the proposed approach.

2 Enterprise Cloud Bus System (ECBS)

Enterprise Cloud Bus System (ECBS) describes a high level abstraction layer of SaaS architecture in Inter-cloud environment, where different clouds interacts and collaborates through cloud agent from various locations in order to publish and/or subscribe their services. The detailed set of building blocks in the context of *ECBS* has been described in Fig. 1.

2.1 Formal Definition of ECBS

In this section, we have discussed about the formal definition of *ECBS* framework. The CloudBus (*CB*) is a set of agents and components and is structurally defined as:

$$CB \rightarrow CLIENT \wedge PA \wedge CUDDI \wedge ESB \wedge CESB \wedge CA \wedge HUDDI \wedge SA \wedge MAPPER \wedge \text{LOGGER} \wedge RES \quad (1)$$

A Multi-cloud environment *Multi-CloudEnv* is that where components of *CB* will work using the following four tuples. It can be formally defined as,

$$Multi - CloudEnv = \{Res, Actor, CB, Relation\} \quad (2)$$

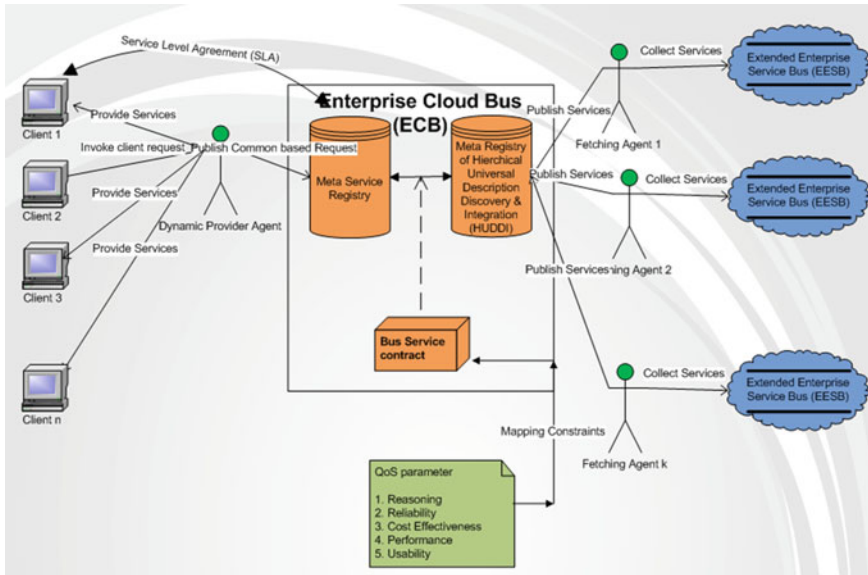


Fig. 1 Enterprise Cloud Bus System Framework (ECBS)

In the given Cloud environment, *Res* is the set of cloud resource, *Actors* are clients of the cloud environment, *CB* are the set of autonomous cloud bus entities with pre specified functions and *Relation* is the set of semantic association and interactions among the cloud bus entities.

2.2 Conceptualization of ECBS in MAS Architecture

A conceptual architecture of ECBS deals with high level representation of the agents and components present inside the cloud architecture. The concept of MAS definition in the proposed architecture is used from [15, 16]. Agent based system are the de facto paradigm to handle the dynamicity of Multi cloud architecture like ECBS.

The dynamicity of CB_i in the environment *Multi-CloudEnv* are handled using three agents $\{PA, SA, CA\}$ and other components relevant to single cloud architecture as described in Fig. 1.

Formally, the dynamic model of any *CloudBus* (CB_i) can be defined as,

$$CB_i = \{CA_i, PA_i, SA_i\} \quad (3)$$

Each of the agents within the CloudBus (CB_i) will be invoked if the following conditions hold by different agents;

$$ESB \wedge CESB \wedge HUDDI \wedge RES \rightarrow CA_i$$

$$CLIENT \wedge CUDDI \wedge RES \rightarrow PA_i$$

$$CUDDI \wedge HUDDI \wedge MAPPER \wedge LOGGER \wedge SCHEDULER \wedge RES \rightarrow SA_i$$

Since, agents are the architectural basis of the (CB_i). Therefore, the dynamic model of the CB_i is a Multi-agent based system and can be defined as a Multi-Agent definition as follows:

$$CB_i \cdot A_i = [Role, E, C, R, PR, K, S, I] \text{ where, } A_i \in \{PA_i, SA_i, CA_i\}.$$

Each agent in the CB_i plays a specific set of Roles in the environment Multi-CloudEnv; *E* is the set of cloud events which occur during various states of the cloud service transitions.

C is a set of environmental conditions in cloud to be checked in order to response on some event in Cloud Bus;

R is a set of environmental cloud resources that are available and necessary for fulfillment of the goal of agents within the CB_i . Formally, ($R \subset Res$);

PR is the properties of agents within the CB_i which will hold the state of the Cloud Bus and also will maintain the state of the Cloud resources *R* on which the agents is acting;

K is the set of information that forms the main knowledge base. Initially it comprises of the states of available cloud resources that the agents within the CB_i will use to response on some event. The K can be update dynamically.

S is the set of cloud services that the agents within the CB_i can provide and conceptualize this will determine the capability of the Cloud Bus components;

I is a set of interactions between the agents reside inside the CB_i .

2.3 ECBS Elements in MAS Architecture

The roles, events and related services along with the respective resources, properties and knowledge base of each agents present within the CB_i is summarized in Table 1.

Here, Provider Agent (PA), Cloud Agent (CA) and Scheduling Agent (SA) starts working with the minimal set of knowledge of the environment to render the request, service and resource token as shown in Table 1. Secondly, all the agents

Table 1 Role collaboration templates of ECBS

Role	Event	Service	Cloud bus component
Agent: Provider Agent (PA)			
R0: Request Transmitter R1: Service Provider	E0: Request Transmitted E1: Service Provided	S0: SendRequest S1: ProvideService	CLIENT
R2: Request Provider	E2: Request Registered	S2: GetRequest	CUDDI
R3: Resource Seeker	E3: Resource used & Released	S3: GetResource S4: ReleaseResource	RES
Agent: Cloud Agent (CA)			
R4: Service Invoker	E4: Service Invoked	S5: InvokeService	ESB
R5: Service Collector	E5: Service Collected	S6: CollectService	CESB
R6: Service Transmitter	E6: Service Registered	S7: PublishService	HUDDI
R7: Resource Seeker	E7: Resource used & Released	S8: GetResource S9: ReleaseResource	RES
Agent: Scheduler Agent (SA)			
R8: Service Matcher	E8: Service Matched	S10: MatchService	CUDDI
R9: Service Seeker	E9: Service Discovered	S11: GetService	HUDDI
R10: Service Mapper	E10: Service Mapped	S12: MapService	MAPPER
R11: Service Scheduler	E11: Service Scheduled	S13: ScheduleService	SCHEDULER
R12: Service Logger	E12: Service Logged	S14: LogService	LOGGER
R13: Service Dispatcher	E13: Service Dispatched	S15: DispatchService	CLIENT
R14: Resource Seeker	E14: Resource used & Released	S16: GetResource S17: ReleaseResource	RES

will use several properties to hold the state of the resources and the states of itself. Finally, the knowledge base is updated dynamically once the component of CloudBus starts working. With all these and with some defined set of Interactions I , the agents will be performing some services like

$$S = \{MatchService, GetService, MapService, ScheduleService, LogService, DispatchService, GetResource, ReleaseResource\}.$$

3 Proposed Enterprise Cloud Bus Petri Net (ECBP)

Petri Nets are a graphical modeling tool that makes a dynamic system more formalized. Therefore, in this section to analyze the behavior of ECBP model we are using a Petri Net tool called PIPE to ensure correctness and performance of the system at design time. The proposed model provides better utilization and automation of cloud resources.

3.1 Definition: Enterprise Cloud Bus Petri Net (ECBP)

A Multi-Agent based Enterprise Cloud Bus Petri Net, *ECBP*, is defined by the 5-tuples as follows:

- (a) $ECBP = [P, T, \Pi, I, O, H, M0, W]$;
- (b) $P = \{P1, P2 \dots, Pm\}$ is a finite set of places.
- (c) $T = \{T1, T2, Tn\}$ is a finite set of transitions, $S \cap T = \emptyset$.
- (d) $I, O, H: T \rightarrow N$ ($N = S \cup T$), are the input, output and inhibition functions.
- (e) $M0$ is the initial marking where, $M0: S \rightarrow IN$ is the initial marking.
- (f) $\Pi: T \rightarrow IN$ is the priority function that associates with the lower priorities to timed transitions and higher priorities to immediate transitions. Immediate transitions therefore always have priority over timed ones.
- (g) W is a weight function $W: F \rightarrow \{1, 2 \dots\}$. $W: T \rightarrow R$ is a function that associates with a real value to the transitions.

The various components of *ECBP* can be mapped to the elements of the conceptual framework as under. The mapping rules are as follows:

A place P in *ECBP* comprises of set of tokens t_k (Request, Service) belongs to constraints, properties, knowledge and services, roles, interactions of any *ECBS* components. Formally, $P \rightarrow t_k$ where, $t_k \in C \cup K \cup PR \cup K \cup S$. All the events E will be mapped as transitions T of a PN. Formally, $T \rightarrow E$.

A component of *ECBS* will request for a resource. Once a resource will be allocated to a component it will hold the resource until the next transition is fired from that component.

The graphical notation of place and transition are represented as Circle and Bar respectively. In *ECBP* it is important to note that, due to firing of any transition *T*, all the sub components of the place *P* will be affected simultaneously. Hence for any place in *ECBP* one can set the mark as 0 or 1.

3.2 *ECBP Elements: Places and Transitions*

The various elements of the proposed *ECBP* will be $\Sigma = [Colour\ set\ for\ request\ token, Colour\ set\ for\ service\ token]$. *C* is the Color function where, $C_r = \{black\ for\ request\ token\}$ and $C_s = \{blue\ for\ service\ token\}$. The color set value of tokens (Request, Service) are marked as $P = 1$; $Q = 2$. The places *P*, transitions *T* and tokens *t* have been summarized in Table 2 and Table 3 respectively.

Table 2 Places and transitions with its descriptions based on Table 1

Places	Component of places	Transitions	Events
P0	Client	T0	E0, E3
P1	PA	T1	E2, E3
P2	CUDDI	T2	E4, E7
P3	ESB	T3	E5, E7
P4	CESB	T4	E6, E7
P5	CA	T5	E8, E14
P6	HUDDI	T6	E9, E14
P7	SA	T7	E10, E14
P8	MAPPER	T8	E11, E14
P9	SCHEDULER	T9	E12, E14
P10	LOGGER	T10	E13, E14

Table 3 Token and its descriptions

Places	Token	Description of tokens	Token parameters	Color set value of token
P0	t0	Request	Sent	1
P1	t0	Request	Provide	1
P2	t0	Request	Register	1
	t1	Service	Register	2
P3	t1	Service	Provide	2
P4	t1	Service	Publish	2
P5	t1	Service	Provide	2
P6	t1	Service	Collect	2
P7	t0	Request	Match	1
	t1	Service	Register	2
P8	t1	Service	Discover	2
P9	t1	Service	Map	2
P10	t1	Service	Schedule	2

4 Analysis of ECBS Based on ECBP

Enterprise Cloud Bus Petri Net (*ECBP*) is a suitable tool to model the behaviour of ECBS system. Moreover, several features of dynamic system like, occurrence of finite number of events, deadlock free operations, achievement of goals through firing of events etc. can be analyzed through the analysis of *ECBP* properties like, safeness, boundedness, liveness, reachability etc.

4.1 ECBP Based Analysis of ECBS

Figure 2 shows the *ECBP* net of ECBS system. The process starts from a place P_0 and after a transition T_1 will reach a place P_1 . The process continues further on and finally arrives at the place P_{10} .

The process from P_0 to P_2 for the transition T_0 to T_2 and from P_3 to P_6 for the transition T_3 to T_6 are occur concurrently because dynamically at the same time clients are registering their request through provider agent in *CUDDI* and services are published by the cloud agent in *HUDDI*. Next, the process is interrupted for service matching with the client request and then from place P_6 it will follow the path of places P_3 , P_7 , P_8 , P_9 and P_{10} for the transition T_3 , T_7 , T_8 , T_9 and T_{10} respectively. If the service is not matched with the required client request the process is cancelled and will be rescheduled.

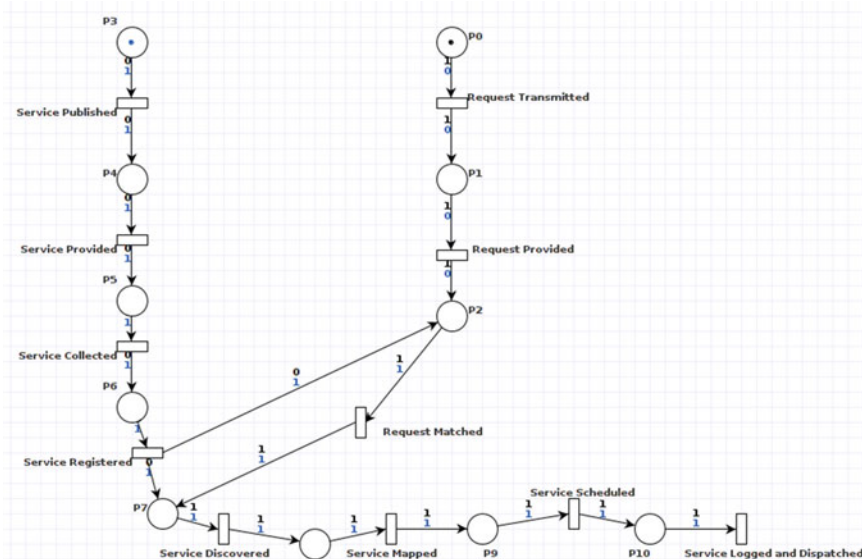


Fig. 2 A Petri Net modeling of Enterprise Cloud Bus System using PIPE: ECBP

4.1.1 Structural Analysis

The structural analysis of the proposed net is shown in Fig. 3. Here, the marking graph is constructed to analyze certain behavioral aspects of the system. We define a siphon as a set of places P satisfying $S \subseteq P$; a siphon is marked by a marking m if at least one of its places is marked by m . In *ECBP* with a siphon S , if S is not marked at the initial marking, then S is not marked at any reachable marking of the proposed system. Similarly, we consider sets of places that never lose all tokens once at least one of their places is marked.

A sufficient condition to siphon is that each transition that removes at least one token from this set also adds a token. This condition is formalized further by the concept of trap. In general, a trap is a set T of places satisfying $T \subseteq T(P)$; a trap is marked by a marking m if at least one of its places is marked at m .

4.1.2 Performance Analysis

The performance analysis of the proposed model is based on the number of reachable tangible markings which is shown in Fig. 4. The corresponding steady-state distribution of tangible states of the model is obtained as follows:

$$X \pi Q = 0 \quad M \in RS \quad \pi[M] = 1;$$

where Q is the infinite simal generator matrix, whose diagonal elements are the rates of the exponential distributions associated with the transitions from one state to other, while the elements on the main diagonal describes the sum of the elements of each row equal to zero. Π is the equilibrium probability mass function (pmf) over the reachable markings M ; usually, we write $\pi[M]$ for the steady-state probability of a given marking M .

Fig. 3 Results for structural analysis of ECBS

State space exploration took 0.651s
Solving the steady state distribution took 0.128s
Total time was 1.359s

Minimal siphons
 {P2_Request_Web Services }

Minimal traps
 {P2_Request }

Analysis time: 0.001s

Marking																							
P7_Request_Web Services		P2_Request_Web Services		P2_Request_Web Services		P0_Request_Web Services		P6_Request_Web Services		P5_Request_Web Services		P10_Request_Web Services		P9_Request_Web Services		P4_Request_Web Services		P3_Request_Web Services		P2_Request_Web Services		P7_Request_Web Services	
Initial	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
Current	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
Enabled transitions																							
Request Matched		Request Provided		Request Transmitted		Service Collected		Service Discovered		Service Logged and Dispatched		Service Mapped		Service Provided		Service Published		Service Registered		Service Scheduled			
no	no	no	no	yes	no	no	no	no	no	no	no	no	no	no	yes	no	no	no	no	no	no	no	
Set of Tangible States																							
P0 P1 P10 P2 P3 P4 P5 P6 P7 P8 P9																							
M0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
M1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
M2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
M3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
M4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	
M5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
M6	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
M7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Steady State Distribution of Tangible States																							
Marking Value																							
M0	-0																						
M1	-0																						
M2	-0																						
M3	-0																						
M4	-0																						
M5	-0																						
M6	-0																						
M7	1																						

Fig. 4 Results for performance analysis of ECBS

Fig. 5 Classification module

State Machine	false
Marked Graph	true
Free Choice Net	true
Extended Free Choice Net	true
Simple Net	true
Extended Simple Net	true

4.1.3 Qualitative and Quantitative Analysis Modules

In this section, we describe different types of qualitative and quantitative analysis modules of Multi-Agent based *ECBS* framework using *ECBP*. The following are the analysis modules of *ECBP*:

- Classification**

This module is based on various places and transitions connectivity of *ECBP* model. We derive the classification module by stating the classification types; State Machine, Marked Graph, FC Nets, *EFC* Nets, *SPL* Nets, and *ESPL* Nets which are shown in Fig. 5.

- GSPN analysis**

Generalized Stochastic Petri Nets (*GSPN*) analysis of *ECBS* using *ECBP* is shown in Fig. 6. This module is characterized by 2 types of transitions:

- Stochastic transitions that is associated with an exponentially distributed firing delay.
- Immediate transitions which are associated with a null firing delay.

Throughput of Timed Transitions		Average Number of Tokens on a Place	
Transition	Throughput	Place	Number of Tokens
Request Matched	0	P0	0
Request Provided	0	P1	0
Request Transmitted	0	P10	0
Service Collected	1	P2	0
Service Discovered	0	P3	0
Service Logged and Dispatched	0	P4	0
Service Mapped	0	P5	0
Service Provided	1	P6	0
Service Published	1	P7	0
Service Registered	1	P8	0
Service Scheduled	0	P9	0

Sojourn times for tangible states		Token Probability Density	
Marking	Value	$\mu=0$	$\mu=1$
M0	0.2	P0	1 0
M1	0.2	P1	1 0
M2	0.2	P10	1 0
M3	0.2	P2	1 0
M4	0.2	P3	1 0
M5	0.2	P4	1 0
M6	0.2	P5	1 0
M7	0.25	P6	1 0
		P7	1 0
		P8	1 0
		P9	1 0

Fig. 6 GSPN analysis

Here, from Fig. 6 we calculate the average number of tokens on a place, the token probability density and the throughput of timed transitions by exploring the state space of the given Petri Net model (*ECBP*) and determining the steady state solution of the model.

• Invariant analysis

In general the occurrences of transitions refer to the transformation of token over a Petri Net. The total number of token for a given set of places remains unchanged if the pre-set and the post-set of the transitions contain the same number of places of this set. The concept of place and transition invariants formalizes such invariant properties.

A vector $w \in Zm$, $w \neq 0$ is a transition invariant of an *ECBP* with m transitions if $Cw = 0$ where C is the incidence matrix of the given Net. A transition invariant describes a group of transitions that may be fired without affecting the marking of the Net. In this section resulted place invariant, transition invariant and place invariant equations are shown in Fig. 7. A vector $v \in Zn$, $v \neq 0$ is a place invariant of an *ECBP* with n places. If $vTM = vTM0$ for all reachable markings of the *ECBP*.

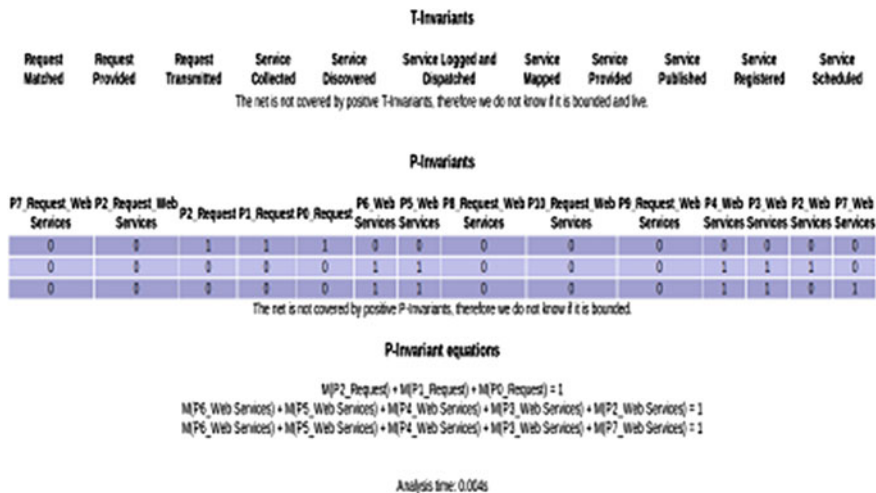


Fig. 7 Invariant analysis result

• Incidence and Marking Matrices of ECBP Model

This module validates the proposed Petri Net model *ECBP* based on the result of the following generated matrices like forward matrix, backward matrix, combined incidence matrices and the inhibition matrix are shown in Fig. 8.

A forward-incidence matrix represents the initial state; backwards-incidence matrix shows the operational state after firing of the set of dynamic events in *ECBS* of combined matrix representing the status of the token at any given instance after the initiation of the token transitions and the inhibition matrix representing the inhibition arcs.

Each of these matrix has been formed using the row constituents $p0_Request$, $p1_Request...p10_Request_Web$ Services and the column constituents Request Provided, Request Transmitted, ..., Service Scheduled. In the *ECBP* model, among the places $P0$, $P1$, $P2$, $P3$, $P4$, $P5$, $P6$, $P7$, $P8$, $P9$, $P10$ none of them are covered and hence the Net is not covered by P invariants. The same is the case for the transitions and the Net is not covered by T invariants.

4.2 Analysis of Behavioral Properties of ECBP Model

Some of the crucial behavioral properties have been analyzed using the *ECBP* model.

- (a) **Safeness:** Any place of a graph is declared as safe, if the number of tokens at that place is either 0 or 1. In the proposed *ECBP*, the set of places $[P1...P10]$ represents a combination of 0 (no token) and 1 (token), which implies that if the

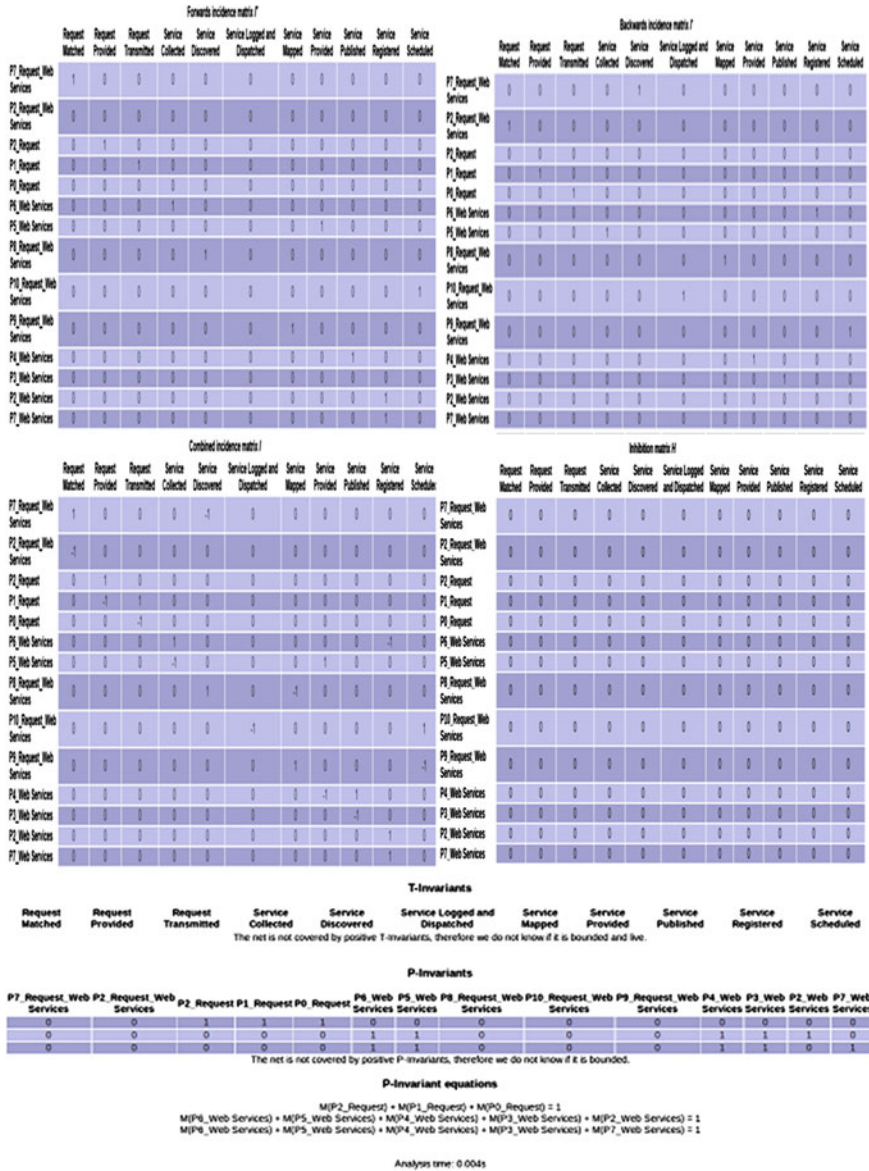


Fig. 8 Incidence and marking matrices of ECBP model

firing occurs there will be a token at the position bit otherwise no token. Thus it shows each of the places has a maximum token count 1 or 0 and is declare safe. Since, all the places in the *ECBP* are safe then the Net as a whole can be declared safe.

- (b) **Boundedness:** The boundedness property specifies the number and type of tokens a place may hold when all reachable markings are considered. The best upper integer bound of a place specifies the maximal number of tokens that can reside on that place in any reachable marking. The best lower integer bound for a place specifies the minimal number of tokens that can reside on that place in any reachable marking. When the best upper and lower integer bounds are equal, this implies that the place always contains a constant number of tokens and thus it is bound. In the proposed net there is no deadlock at any stage within $P1$ to $P10$ and hence it is bounded. The proposed *ECBP* Net is safe and so the number of tokens in a place is restricted to either 0 or 1.
- (c) **Reachability:** Reachability Coverability graph in Fig. 9 provides a pictorial representation of all the possible transitions firing sequences of *ECBP* model. The graph is used to define a given Petri Net N and marking M , where M belongs to $R(N)$. The nodes of the graph are identified as markings of the Net $R(N, M0)$, where $M0$ is the initial marking and the arcs are represented by the transitions of N . Each initial marking $M0$ has an associated Reachability set. This set consists of all the markings that can be reached from $M0$ through the firing of one or more transitions. In our proposed model the reachability

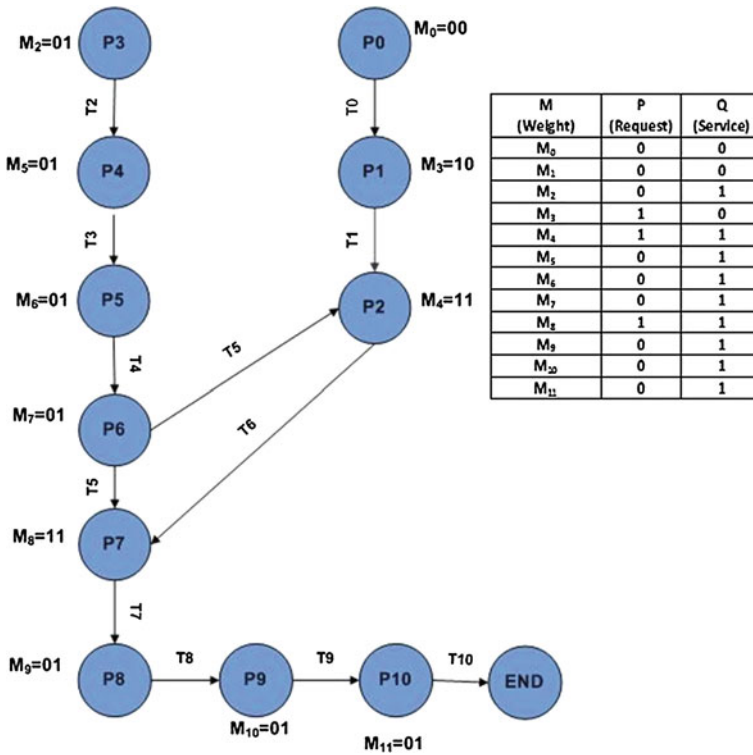


Fig. 9 Reachability coverability graph of ECBP

coverability graph starts with initial marking $M0 = [0\ 0]\ T$ and finally reach to state $M11 = [0\ 1]\ T$, where we conclude the service session for the current request of shutdown. As stated earlier, the mark of each place in P of *ECBP* model will be treated as 1 or 0.

- (d) **Liveness:** In the proposed *ECBP* model as the process starts from $P0$ transitions $T0$ through $T9$ are fired and place $P10$ is reached. The Net is continuous and hence the liveness property is ensured. Therefore the proposed Net is considered to be live.
- (e) **Conservativeness:** Conservation property of *ECBP* model defines that the number of tokens before and after the execution remains constant. Here, sum of all tokens is counted at the initial markings and again in the final markings of each execution. If all the markings in the reachability graph have the same sum of tokens then the Petri Net is declared to be strictly conservative. Hence, *ECBP* is considered to be strictly conservative.
- (f) **Fairness:** Fairness is an important performance criterion for studying behavioral properties of dynamic system. The fairness index always lies between 0 and 1. Here, we have discussed about the bounded fairness or *B Fair*. An *ECBP* Net is said to be *B Fair* Net where every pair of transitions in the Net are in a *B* fair relation.

5 A Case Study

In this work, we establish our approach with the help of a case study of an airline reservation system. Here we provide the Petri Net modeling of *ECBS* using a tool called PIPE. This application is used to maintain flight details, flight status and reservation process. This Petri Net model can be used by other service industries as well.

Major features provided by the system are:

- (a) **Enquiry about the flight details:** The airline reservation system allows the user to perform flight inquiry such as flight scheduling, availability, status, fare details, etc.
- (b) **User Registration:** The system allows the user to register as a new user who can book.
- (c) **Reservation of Flight:** The system allows the user to book the flights as per their requirements.

Using Tables 4 and 5 we have modeled the roles in the Airline Reservation System that are shown in Fig. 10. The following are the workflow of the proposed case study:

Table 4 Places and transitions with its descriptions

Places	Component of places	Transitions	Events
P0	Customer	T0	E0
P1	Booking Portal	T1	E1
P2	Airline Server	T2	E2
P3	Airline Server	T3	E3
P4	Airline Agent	T4	E4
P5	Flight Details	T5	E5
P6	Airline Reservation Gateway	T6	E6
P7	Flight Scheduling Agent	T7	E6
P8	Flight Reservation Mapper	T8	E7
P9	Flight Reservation Scheduler	T9	E8
P10	Airline Reservation Details	T10	E9

Table 5 Transitions with its descriptions

Places	Events details	Token	Description of tokens	Token parameters
P0	Customer Request Transmitted	t0	Request	Sent
P1	Customer Request Provided	t0	Request	Provided
P2	Flight Details Published	t0 t1	Request Service	Registered Registered
P3	Flight Details Provided	t1	Service	Provided
P4	Flight Details Collected	t1	Service	Published
P5	Flight Details Registered	t1	Service	Provided
P6	Flight Details Matched	t1	Service	Collected
P7	Flight Details Discovered	t0 t1	Request Service	Matched Registered
P8	Flight Details Mapped	t1	Service	Discovered
P9	Flight Details Scheduled	t1	Service	Mapped
P10	Flight Details Logged & Dispatched	t1	Service	Scheduled

(1) End Users:

- Initially, a Customer requests for booking the ticket according to their respective requirements such as source, destination, date, time and the price range.
- These request parameters are passes through the Booking Portal that acts as a provider agent and are registered on Airline Reservation system Gateway.
- Airline Reservation Gateway acts as a *CUDI* (Cloud Universal Description Discovery and Integration) which is considered to be an end customer.

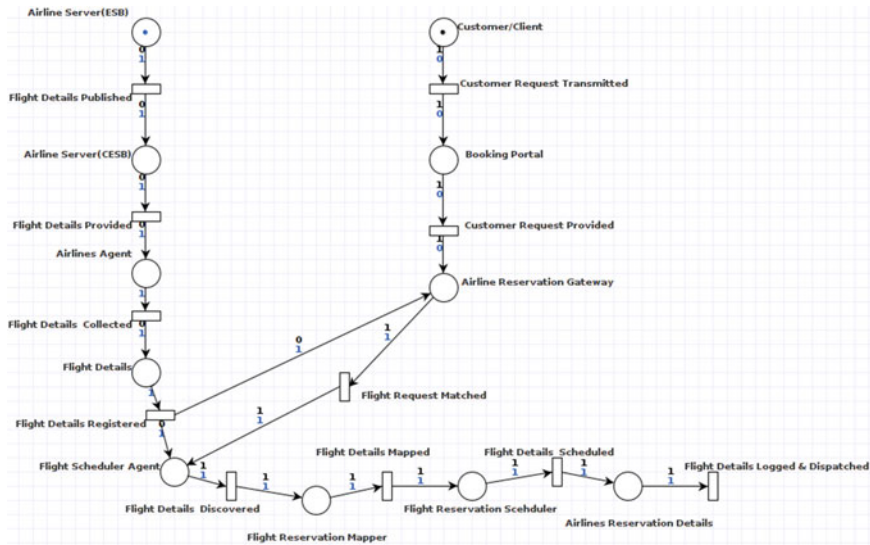


Fig. 10 Modeling of airline reservation system using ECBP model

(2) End Service Provider

- (a) In this architecture an Airline Server which acts as an *ESB* (Enterprise Service Bus) is shown that contains the database of the entire flights that keeps track of the flights and is regularly updated whether new flights are introduced or existing flights are cancelled.
- (b) Another airline server which acts as a *CESB* (Cloud Enterprise Service Bus) is the extension of *ESB* that keeps the record of those flights which are running on the particular day.
- (c) The Airlines Agent which acts like a cloud agent maintains the record of number of seats booked and seats available according to the flight status.
- (d) The entire information is registered under the Flight Details which works as an *HUDDI*. Here, the matching for the required flights is done.
- (e) The Flight Scheduler agent act as a scheduler agent that keeps track the flight schedule.
- (f) Flight Reservation Mapper acts as a Service Mapper which maps the request parameters with the available seats to find the best results.
- (g) Then Flight Reservation Scheduler schedules the required flight as per the customer's choice.
- (h) Finally the Airlines Reservation Details which act as a Service logger will book the best available flight according to the customer's request.

6 Conclusion

In this work, the main focus is to formalize the *ECBS* and established the conceptual definition of *ECBS* in *MAS* architecture. Since, *ECBS* is considered to be a highly dynamic in nature in terms of its components interactions with the environments, handling of environmental resources and activities to achieve the pre specified goal. Therefore, we designed the dynamics of *ECBS* using a Petri Net tool called *PIPE*. The key benefits of using Petri Net modeling of *ECBS* dynamics are high accuracy of system functionality, operational design of system flexibility that comprises of autonomous agents and system components, automation of cloud services and quantitative measurement of system processes. Further, through *ECBP* concepts and corresponding reachability graph, several important behavioral properties of *ECBS* like, safeness, boundedness, liveness, etc. are analyzed. A case study has also been explained based on the proposed approach.

Future work includes, the development of a more expressive behavioral analysis model of *ECBP* using High Level Petri Net because using Petri Net approach tokens cannot be distinguishable to analyze the complex Net easier when complexity of the Net increases with the increase of extended properties. The limitations of the proposed model are inability to support similar kind of service interactions in one single net and less expressive in analyzing the behavioral properties of *ECBS* such as safeness, boundedness and liveness, etc. in compare to High Level Petri Net.

References

1. Huhns, M.N., Singh, M.P.: Service-oriented computing: key concepts and principles. *IEEE Internet Comput.* **9**(1), 75–81 (2005)
2. Arsanjani, A.: Service oriented modeling and architecture. *IBM developer works*, pp. 1–15 (2004)
3. Zheng, Z., Lyu, M.R.: Collaborative reliability prediction of service-oriented systems. In: *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering*, vol. 1, pp. 35–44 (2010)
4. Papazoglou, M.P., Van Den Heuvel, W.J.: Service-oriented computing: state-of-the-art and open research issues. *IEEE Comput.* (2003)
5. Alexandros, K., Aggelos, G., Vassilis, S., Lampros, K., Magdalinos, P., Antoniou, E., Politopoulou, Z.: A cloud-based farm management system: architecture and implementation. *J. Comput. Electron. Agric.* **100**, 168–179 (2014)
6. Cavalcante, E.: Architecture Driven Engineering of Cloud-Based Applications. *IFIP/ Springer-Verlag, Germany*, vol. 22, pp. 175–180 (2013)
7. Buyya, R., Ranjan, R., Rodrigo, N.: Intercloud: utility-oriented federation of cloud computing environments for scaling of application services. In: *Algorithms and architectures for parallel processing*, vol. 6081, pp. 13–31. *LNCS. Springer* (2010)
8. Brandtzæg, E., Parastoo, M., Mosser, E.: Towards a domain-specific language to deploy applications in the clouds. In: *3rd International Conference on Cloud Computing, GRIDs, and Virtualization. IARIA*, pp. 213–218 (2012)

9. Ferry, N., Alessandro, R., Chauvel, F., Morin, B., Solberg, A.: Towards model-driven provisioning, deployment, monitoring, and adaptation of multi-cloud systems. In: 2013 IEEE Sixth International Conference on cloud computing, pp. 887–894 (2013)
10. Divyakant, A., Abbadi, A., Das, S., Elmore, A.J.: Database scalability, elasticity, and autonomy in the cloud. *J. Database Syst. Adv. Appl.* **2**, 2–15 (2011)
11. Zambonelli, F., Omicini, A.: Challenges and research directions in agent-oriented software engineering. *J. Auton. Agents Multi-Agent Syst.* **9**, 253–283 (2004)
12. Bauer, B., Muller, J.P., Odell, J.: Agent UML: a formalism for specifying multiagent software systems. *Int. J. Softw. Eng. Knowl. Eng.* **11**(3), 1–24 (2001)
13. Klügl, F.: Measuring Complexity of multi-agent simulations—an attempt using metrics. In: *Languages, Methodologies and Development Tools for Multi-Agent Systems*. Springer, Berlin, Heidelberg (2008)
14. Dhavachelvan, P., Saravanan, S., Satheskumar, K.: Validation of complexity metrics of agent-based systems using Weyuker’s axioms. In: *International Conference on Information Technology (ICIT ’08)*, pp. 248–251 (2008)
15. Sarkar, A., Debnath, N.C.: Measuring complexity of multi-agent system architecture. In: 10th IEEE Conference on Industrial Informatics, pp. 998–1003 (2012)
16. Sarkar, A.: Modeling multi-agent system dynamics: graph semantic based approach. In: 10th International Conference on Service Systems and Service Management, pp. 664–669 (2013)
17. Djamel, B.: An agent-based approach for hybrid multi-cloud applications. *J. Scalable Comput. Pract. Experience* **2**(14), 95–109 (2013)
18. Mandal, A., Changder, S., Sarkar, A., Debnath, N.: A novel and flexible cloud architecture for data-centric applications. In: *IEEE International Conference on Industrial Technology (ICIT)*, pp. 1834–1839 (2013)
19. Khan, G., Sengupta, S., Sarkar, A., Debnath, N.C.: Modeling of inter-cloud architecture using UML 2.0: multi-agent abstraction based approach. In: 23rd International Conference on Software Engineering and Data Engineering, pp. 149–154 (2014)
20. Khan, G., Sengupta, S., Sarkar, A.: Modelling of services and their collaboration in Enterprise Cloud Bus (ECB) using UML 2.0, 2015. In: *International Conference on Advances in Computer Engineering and Applications*, pp. 207–213, India (2015)
21. Khan, G., Sengupta, S., Sarkar, A.: WSRM: a relational model for web service discovery in Enterprise Cloud Bus (ECB). In: 3rd International Conference on Eco-friendly Computing and Communication System, pp. 117–122, India (2014)
22. Khan, G., Sengupta, S., Sarkar, A., Debnath, N.C.: Web service discovery in enterprise cloud bus framework: T vector based model. In: 13th IEEE International Conference on Industrial Informatics, pp. 1672–1677 (2015)
23. Khan, G., Sengupta, S., Sarkar, A., Debnath, N.C.: XML based service registration system for enterprise cloud bus. In: 3rd IEEE International Conference on Computing, Management and Communications, pp. 250–255 (2015)
24. Khan, G., Sengupta, S., Sarkar, A.: Priority based service scheduling in enterprise cloud bus architecture. In: *IEEE International Conference on Foundations and Frontiers in Computer, Communication and Electrical Engineering (C2E2 2016)*, SKFGI, pp. 363–368, Mankundu, India (2016)
25. Khan, G., Sengupta, S., Sarkar, A., Debnath, N.C.: Performance analysis of service discovery for Enterprise Cloud Bus (ECB). In: *IEEE International Conference on Industrial Technology (ICIT 2016)*, pp. 1728–1735, Taipei, Taiwan (2016)
26. Bloom, J., Clark, C., Clifford, C., Duncan, A., Khan, H., Papantoniou, M.: *Platform Independent Petri-Net Editor: Final Report*, London (2003)
27. BoNet, P., Llado, M.C., Puigjaner, R.: PIPE v2.5: a Petri Net tool for performance modeling. In: *Proceedings of the 23rd Latin American Conference on Informatics (CLEI 2007)*, San Jose, Costa Rica (2007)
28. Marzougui, B., Hassine, K., Barkaoui, K.: A new formalism for modeling a multi-agent systems: agent Petri Nets. *J. Softw. Eng. Appl.* **3**(12), 1118–1124 (2010)

29. Celaya, J.R., Desrochers, A.A., Graves, R.J.: Modeling and analysis of multi-agent systems using Petri Nets. *J. Comput. Academy Press* **4**(10), 981–996 (2009)
30. Chainbi, W.: Multi-agent systems: a Petri Net with objects based approach. In: *IEEE/WIC/ACM International Conference on Intelligent Agent Technology* (2004)
31. Pujari, S., Mukhopadhyay, S.: Petri Net: a tool for modeling and analyze multi-agent oriented systems. *Int. J. Intell. Syst Appl.* **10**, 103–112 (2012)
32. Chatterjee, A.K., Sarkar, A., Bhattacharya, S.: Modeling and analysis of agent oriented system: Petri-net based approach. In: *11th International Conference on Software Engineering Research and Practice (SERP 11)*, vol. 1, pp. 17–23 (2011)
33. Sofiane, B., Bendoukha, H., Moldt, H.: ICNETS: towards designing inter-cloud workflow management systems by Petri Nets. In: *Enterprise and Organizational Modeling and Simulation*, vol. 198, pp. 187–198. Springer International Publishing (2015)

Advanced Computing and Systems for Security

Volume Four

Chaki, R.; Saeed, K.; Cortesi, A.; Chaki, N. (Eds.)

2017, XI, 179 p. 66 illus., Softcover

ISBN: 978-981-10-3390-2