International Journal of Software Engineering and Its Application
Vol.11, No.7 (2017), pp.13-30
http://dx.doi.org/10.14257/ijseia.2017.11.7.02

Behavioral Modeling of Enterprise Cloud Bus System: High Level Petri Net based Approach

Gitosree Khan[1], Anirban Sarkar[2] and Sabnam Sengupta[1]

[1]Department of Information Technology, B. P. Poddar Institute of Management & Technology, Kolkata, India
[2]Department of Computer Applications, National Institute of Technology, Durgapur, India
{khan.gitosree, sarkar.anirban, sabnamsg}@gmail.com

Abstract

In recent days number of Enterprise Software Applications is increased rapidly with the increasing number of clouds and its services. Therefore, the need for enterprises to leverage cloud services dynamically as "on-demand" basis is growing exponentially. In this work, a conceptual architecture of Multi-agent based Enterprise Cloud Bus System (ECBS) is proposed and modelled its dynamics using High Level Petri Net (HECBP) based approach. The proposed approach is beneficial for the cloud based enterprise applications in optimizing the performance, cost, elasticity, flexibility, high reliability and availability of the computing resource. The proposed mechanism is capable of modelling and analyzing the behavioral facets of Enterprise Cloud Bus which are structured based on Multi-agent based system (MAS) and the behavioral features of inter-cloud architecture. Using the HECBP concepts and corresponding reachability graph, several key behavioral properties of cloud based systems like, reachability, safeness, Boundedness, liveness can be analyzed formally. The proposed HECBP is simulated using Colored Petri Net tools.

Keywords*: Cloud computing, Enterprise cloud Bus, Multi-agent system, Behavioral analysis, High level Petri-net, Colored Petri-net*

1. Introduction

The growing complexity of Enterprise Software Applications and the increasing number of clouds throughout the world have increased the challenges for Software as a Service in the recent trends of Software and Requirement Engineering. Thus the demand for service and cloud computing technology [1], [2] towards organizations is growing exponentially. Lots of researchers [3], [4] discusses over the architectural design of cloud computing and its applications. Among them [5],[6] focus on the architecture and implementation of Inter-cloud applications that facilitates monitoring cloud services, composing, and adapting cloud applications. But still, there exists a problem in formalization of Multi-cloud architecture. In recent days, several researches in last decade, have devised conceptual model for Multi-cloud architecture which aim at addressing the issues. The authors in [7], [8] focuses on the Model-driven approach for dynamic provisioning and deployment of Inter-cloud architecture. The authors in [9] discusses about the latest challenges of architectural components in Multi-cloud system. But, all these approaches have got certain limitations to exhibit the dynamism of internal behavior of the system which comprises of heterogeneous set of components.

In this context, analysis of such dynamics is a major challenge. For this purpose, proper mechanism is required to conceptualize and study the behavioral properties of Multi-cloud based architecture. Agent-oriented systems are the most acceptable paradigm to handle the dynamicity of Inter-cloud architecture. Formalization and analysis of dynamic

ISSN: 1738-9984 IJSEIA

facets of Multi-agent system has been explained in paper [10]. Few of the research works [11], [12] are based on measurement and validation of complexity metrics. Moreover, modeling and design of Agent-based approach of Multi-cloud architecture [13] has emerged as one of the most challenging domains in cloud computing domain. As our previous work are based on such MAS based Inter-cloud architecture [14], called Enterprise Cloud Bus (ECB).Few of our earlier work are based on service registration and discovery mechanism in ECBS [15], [16] which helps to identify services during run time. Further, in [17], [18] scheduling and composition of web services in Multi- cloud environment has been discussed by the author. 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 is a Platform Independent Petri Net Editor Tool that helps to model and analyze the dynamics facets of any Multi-agent cloud architecture [19]. The authors in [20], [21] discusses about the modeling and analysis of the dynamic facets of MAS using Petri-net based approach. Moreover, few of the researchers work on the modeling and analysis of Inter-cloud architecture [22], [23] using Petri-Net based approach. However, those approaches are also less expressive for Inter-cloud architecture comprising of multiple agents and components. Therefore, Colored Petri Net (CPN) Tools is an efficient tool [24] that is used for constructing and analyzing such Multi-cloud system. Many of the work reveals about the behavioral analysis of Multi-cloud architecture using Colored Petri Net. But they are some shortcomings towards analysis of dynamics Multi-cloud architecture. Therefore, High level Petri Net based approach [25] is most suitable towards analysis of dynamics of such system.

This paper is the extension of the work done in [23], which handles the dynamic facets and behavioral modeling of complex MAS based Inter-cloud architecture (ECBS) using a Petri-net tool called PIPE. This paper particularly focuses on the formal definition of ECBS that conceptually defined in MAS environment. The main objective of this paper is to analyze the dynamic properties of ECBS using High level Petri Net tool called CPN. Further, through the proposed HECBP net and reachability graph, the behavioral properties of ECBS like safeness, liveliness, boundedness, etc are analyzed. The proposed model is effective towards interactions among the heterogeneous agents present within the cloud bus.

2. Enterprise Cloud Bus System (ECBS)

An Enterprise Cloud Bus System (ECBS) is a hierarchical layered of SaaS architecture in Inter-cloud environment, where various clouds from different locations interact and collaborate among each other to publish their services. Figure 1 shows the Enterprise Cloud Bus System with its components. The detailed sets of building blocks of ECBS have been described in [14].

2.1. Formalization of ECBS

This paper is the extensions of ESB's with formal approach towards analysis of dynamics of Inter-cloud architecture *ECBS*. The *CloudBus* (*CB*) is the set of agents and components (as refer in figure 1) of Enterprise Cloud Bus System framework.

The structural representation of the *CloudBus* (*CB*) is defined as:

$$CB \xrightarrow{yields} CLIENT \cap PA \cap CUDDI \cap ESB \cap CESB \cap CA \cap HUDDI \cap SA \cap MAPPER \cap LOGGER \cap RES \quad (1)$$

A Multi-cloud environment *Multi-CloudEnv* is that where components of *CB* will work using the following four tuples.

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

In the given Cloud scenario, *Res* refers to the cloud resource shared by the cloud bus, Actors are clients of the cloud environment, *CB* refers to the set of cloud bus entities with pre specified functions and Relation refers to group of association and interactions with meaningful semantics among the cloud bus entities. In the context of *Multi-CloudEnv*, the *CB* will comprehend each state of the events and response according to its cloud services. Further, Enterprise Cloud Bus System (*ECBS*) can be defined as:

$ECBS = \{CB, COL, I\}$, where, *CB* is the set of Cloud Bus.

Thus, $\forall i$, $CB_i \in CB$. COL_{ij} identifies a set of Collaborations among CB_i and CB_j. Thus, $\forall i, j$, $COL_{ij} \in COL$, if $i \neq j$. The set I_{ij} identifies the interaction path between any two Cloud Bus CB_i and CB_j in the *ECBS* system. Thus, $\forall i, j$, $I_{ij} \in I$, if $i \neq j$.

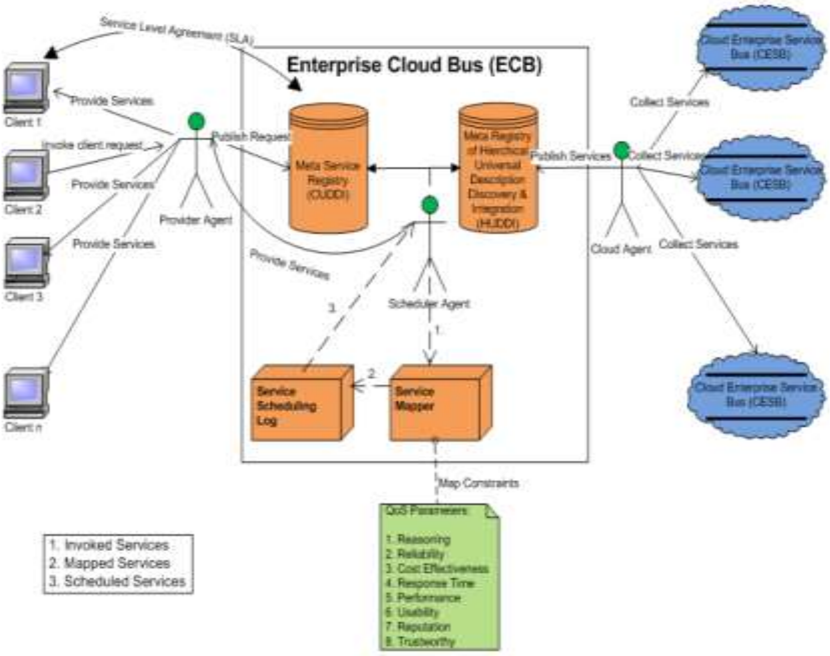


Figure 1. Enterprise Cloud Bus System

2.2. Conceptualization of ECBS in MAS Architecture

Conceptual architecture of *ECBS* defines a set of building blocks and their interconnections to define conceptually the environmental entities, agents, collaborations and interactions among the *ECBS* and high level representation of the agents and other components in Inter-cloud architecture.

This section describes the conceptual definition of Multi-agent based Enterprise Cloud Bus System (*ECBS*). The concept of *MAS* definition in the proposed architecture is used from [21], [23], [25].

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 Figure 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;

$$ESB \cap CESB \cap CA \cap HUDDI \cap RES \xrightarrow{yields} CA_i$$

$$CLIENT \cap CUDDI \cap RES \xrightarrow{yields} PA_i$$

$$CUDDI \cap HUDDI \cap MAPPER \cap LOGGER \cap SCHEDULER \cap RES \xrightarrow{yields} SA_i$$

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

CB_i . A_i= *[Role, E, C, R, PR, K, S, I]* where, $A_i \in \{PA_i, SA_i, CA_i\}$ where, each agent in the CB_i plays a specific set of Roles in the environment *Multi-CloudEnv* and *E* is the set of cloud events which occur during various states of the cloud service transitions, *C* refers to the set of conditions in a given cloud environment that is checked before response on some other event in Cloud Bus, *R* refers to a set of environmental cloud resource that are necessary to fulfill the goal of agents on demand within the CB_i.

Formally, ($R \subset Res$), *PR* refers to the properties of agents within the CB_i which will hold the status of the *CloudBus* and the Cloud resources *R* over which the agents is acting, *K* identifies the set of information that builds the main knowledge base. At the initial stage, *K* comprises of the states of *R* that cloud agents will use to response on some other event. Here, *K* can be update automatically, whereas, *S* refers the set of cloud services an agents can provide dynamically and further conceptualize to determine the capability of the cloud bus components; *I* refers to a set of interaction path between the agents reside inside the CB_i;

2.3. Structural Analysis of ECBS

The structural analysis of the *ECBS* system can be studied using equations (1), (2) and (3) as described in the earlier section. The analysis states that, CA_i exists if for all *i*, *ESB*, *CESB*, *HUDDI* and *RES* components exist. This also implies that for any CloudBus *i*, any change in CA_i will affect the state of *ESB*, *CESB*, *HUDDI*, *RES* only.

Similarly, PA_i exists if for all *i*, *CLIENT*, *CUDDI* and *RES* exist, which implies for any CloudBus *i*, any change in PA_i will affect the state of *CLIENT*, *CUDDI*, *RES* only. Similarly, SA_i exists if for all *i*, *CUDDI*, *HUDDI*, *MAPPER*, *LOGGER*, *SCHEDULER* and *RES* components exist. That implies, for any *CloudBus i*, change in SA_i will affect the state of *CUDDI*, *HUDDI*, *MAPPER*, *LOGGER*, *SCHEDULER*, and *RES* only.

2.4. ECBS Elements in MAS Architecture

The roles, events and related services along with the respective resources, properties and knowledge base of *PA*, *CA* and *SA* present within the CB_i is summarized in Table 1. Provider Agent (*PA*) starts working with the minimal set of knowledge of the environment to render the request, service and resource token. The set of Roles *R* as shown in Table 1 to be played by the *PA* will be

R= {R0: Request Transmitter, R1: Service Provider, R2: Request Provider, R3: Resource Seeker}.The set of Events *E* for the *PA* will be *E= {E0: Request Transmitted, E1: Service Provided, E2: Request Registered, E3: Resource used & Released}*.These set of events will be performed after satisfying possible environmental constraints *C*. The set Resources *RS = {RS1: Web Service, RS2: Registries, RS3: Timestamp}*. Now the *PA* will use several properties to hold the state of the resources and the states of itself. Hence the set of Properties

PR = {PR1: status of Request type which can have various statuses, PR2: status of Service type which can have various statuses, PR3: status of Resource type which can have various statuses}.

PA starts working with the minimal set of knowledge of the environment to render the services. The knowledge base can be updated dynamically once the component of CloudBus starts working. The set of knowledge

K = {K0: Details of request, K1: Details of service, K2: Details of resource}. With all these and with some defined set of Interactions I PA will be performing some services

S = {S0: SendRequest, S1: ProvideService, S2: GetRequest, S3: GetResource, S4: ReleaseResource}.

Cloud Agent (*CA*) starts working with the minimal set of knowledge of the environment to render the request, service and resource token. The set of Roles *R* as shown in Table 2 to be played by the *CA* will be

*R= {R4: Service Invoker, R5: Service Collector, R6: Service Transmitter, R7: Resource Seeker}.*The set of Events *E* for the *CA* will be *E= {E4: Service Invoked, E5: Service Collected, E6: Service Registered, E7: Resource used & Released}.*

These set of events will be performed after satisfying possible environmental constraints *C*. The set of Resources *RS = {RS1: Web Service, RS2: Registries, RS3: Timestamp}.*Now the *CA* will use several properties to hold the state of the resources and the states of itself. Hence the set of Properties

PR = {PR4: status of Request type which can have various statuses, PR5: status of Service type which can have various statuses, PR6: status of Resource type which can have various statuses}.

CA starts working with the minimal set of knowledge of the environment to render the services.

The knowledgebase can be updated dynamically once the component of *CloudBus* starts working. The set of knowledge

*K = {K3: Details of request, K4: Details of service, K5: Details of resource}.*With all these and with some defined set of Interactions I, CA will be performing some services

S = {S5: InvokeService, S6: CollectService, S7: PublishService, S8: GetResource, S9: ReleaseResource}.

Scheduling Agent (*SA*) starts working with the minimal set of knowledge of the environment to render the request, service and resource token. The set of Roles R as shown in Table 2 to be played by the *SA* will be

R= {R8: Service Matcher, R9: Service Seeker, R10: Service Mapper, R11: Service Scheduler, R12: Service Scheduler; R13: Service Logger; R14: Resource Seeker}.

The set of Events *E* for the *SA* will be *E= {E8: Service Matched, E9: Service Discovered, E10: Service Mapped, E11: Service Scheduled, E12: Service Logged, E13: Service Dispatched E14: Resource used &released}.*

These set of events will be performed after satisfying possible environmental constraints *C*. The set of Resources *RS = {RS1: Web Service, RS2: Registries, RS3: Timestamp}.*Now the *SA* will use several properties to hold the state of the resources and the states of itself. Hence the set of Properties

PR = {PR7: status of Request type which can have various statuses, PR8: status of Service type which can have various statuses, PR9: status of Resource type which can have various statuses}.

SA starts working with the minimal set of knowledge of the environment to render the services. The knowledgebase can be updated dynamically once the component of CloudBus starts working. The set of knowledge *K = {K6: Details of request, K7: Details of service, K8: Details of resource}.* With all these and with some defined set of Interactions I, SA will be performing some services

S= {MatchService, GetService, MapService, ScheduleService, LogService, DispatchService, GetService, ReleaseService}.

Table 1. Role Collaborations Template for 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:ReleseResource	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:ReleseResource	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:ReleseResource	RES

3. Proposed High Level Enterprise Cloud Bus Petri- Net

High-level Enterprise Cloud Bus Petri Nets (*HECBP*) refers to a graphical representation of *ECBS* architecture that allows visualization and analysis of the system dynamics and behavioral characteristics such as safeness, boundedness, liveliness, etc. Proposed HEC*BP* is a Colored Petri Net (*CPN*) based approach which is capable to represent the interactions between agents and other cloud components within the cloud bus.

3.1. Definition: High Level Enterprise Cloud Bus Petri Net (HECBP)

A High Level Petri net is defined as a directed bipartite graph which consists of two nodes namely places and transitions. The arc that connects the nodes represents state of a transition of the given node.

Hence in a formal manner, High Level Enterprise Cloud Bus Petri Net, (*HECBP*), is described by the 8 – tuples as follows:

$HECBP = \{\Sigma, P, T, N, C_f, G, E, I\}$

The various elements of the proposed *HECBP* is defined as, *Σ = [Colour set for Request token, Colour set for Service token, Colour set for Resource token]*. C_f is the colour function where, C_{re} *= {blue for request}*, C_s *= {red for service}*, C_r *= {black for resource token}.*

Σ refers to a finite set of non-empty colour sets. The various types of such colour sets determines the data values of *CloudBus* components, resources, operations and functions that are used in the net expressions, $\Sigma = .\ C_s \cup C_{re} \cup C_r \cup G \cup E \cup I;$

P is a non-empty finite set of places. It comprises of all the *CloudBus* and their environmental resources.

Formally, $P = \sum_{i=1}^{n} p\ CB_i \ \cup \sum_{j=1}^{n} p\ Res$

The *CloudBus* place contains all the agents, components, tokens, except events, of a *CloudBus*. *T* is a non-empty finite set of transitions include all events of any *CloudBus*, CB_i and resource R_i, along with the interactions between the Cloud Bus present in environment, $T = CB_i \cup I \cup R_i * e_i$.. *N* is the finite set of arcs that map to a pair between source node and the destination node. The two nodes are of different nature. If we say that, *T = e U I*, then it can be said that, $T \times P\ (CB_i)\ U\ (P \times T)$, because arc from *T* to *P* is not valid in case of resources. Hence,

C_f is the color function. Formally, $C_f \rightarrow C_b$. The color function C_f maps each place *P* to a type *C*. *C* is the color function for *CloudBus* that contains various tokens of different colors. $C_b = C_s \cup C_{re} \cup C_r$, C_s is the color function for Service token, C_{re}, is the color function for Request token and C_r is the color function for resources. Different tokens have different colors in the Net.

The guard function of the net expressions *G* maps each transition, *T* into a Boolean expression in which all variables of colour set types belongs to *Σ*. The arc expression function *E* of the net maps each arc '*a*' in the node function into an expression of type C_f *(p)*.This means that each arc expression must evaluate to multi-set over the type of the adjacent place, *P*. The initialization function *I* map each place, *P*, into a Boolean expression of type $C_f\,(p)$.

An agent within the cloud bus of *ECBS* comprises of various elements namely roles, events, constraints, resources, properties, knowledge, interactions and services which together make *ECBS* successful to achieve the pre specified goal. Mapping of Conceptual Architectural to *HECBP* has been summarized in Table 2. A component 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 place.

Formally, $CB_i \rightarrow RES$. The graphical notation of place and transition are represented as usual notation of CPN and those are Circle and Bar respectively.

Table 2. Mapping from ECBS Conceptual Architecture to HECBP

Serial No.	Concepts in ECBS	Concept in HECBP
1	Properties, Knowledge, Cloud Services, Roles of (CLIENT, PA, CUDDI, ESB, CESB, CA, HUDDI, SA, MAPPER, LOGGER, RES)	Place, P
2	Events of (CLIENT, PA, CUDDI, ESB, CESB, CA, HUDDI, SA, MAPPER, LOGGER, RES)	Transitions, T
3	Collaborations among (PA, SA, CA)	Set of Arcs, N
4	Elements of (PA, SA, CA)	Color Function, C_f
5	Constraints of (PA, SA, CA)	Guard Function, G
6	Interactions among (PA, SA, CA)	Arc Expression, Exp
7	Users	Initialization Function, I

3.2. HECBP Elements: Places & Transitions

The details of the places *P*, transitions *T* and tokens t have been illustrated in Table 3 and Table 4 respectively. In the proposed *HECBP* model we have considered three types of tokens namely service, request and resource token. All the tokens are distinguished among themselves by their color used in the net (Red for Service token, Blue for Request token, Black for Resource token).

Table 3. Places & Transitions with its Descriptions based on Table 1 & 2

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
P11	RESOURCE (RS1, RS2, RS3) RS1: Web Services; RS2: Registries;RS3: Timestamp	T11	E1,E14

Table 3 represents descriptions of the transitions and events of the corresponding places. Table 4 shows the description of the tokens and its parameters that are used in our work. The color set value of the tokens is considered as (*P=1; Q=2; R=3*) to distinguish among themselves;

Table 4. Token and its Descriptions

Places	Token	Description of Tokens	Token Parameters	Colour Set value of Token
P0	t0	Request	Sent	1
P1	t0	Request	Provide	1
P2	t0 t1	Request Service	Register Register	1 2
P3	t1	Service	Published	2
P4	t1	Service	Provide	2
P5	t1	Service	Collect	2
P6	t1	Service	Register	2
P7	t1 t1	Service Service	Register Dispatch	2 2
P8	t1	Service	Discover	2
P9	t1	Service	Map	2
P10	t1	Service	Schedule	2
P11	t2 t2	Resource Resource	Request Release	3 3

4. Analysis of ECBS based on HECBP

High Level Enterprise Cloud Bus Petri Net (*HECBP*) is a suitable Petri-net 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 *HECBP* properties like, safeness, boundedness, liveness, reachability etc. Further, the *HECBP* based analysis will give detail insight about the internal behaviour of the system.

4.1. HECBP based Analysis of ECBS

The *HECBP* net of the *ECBS* framework is shown in figure 2.

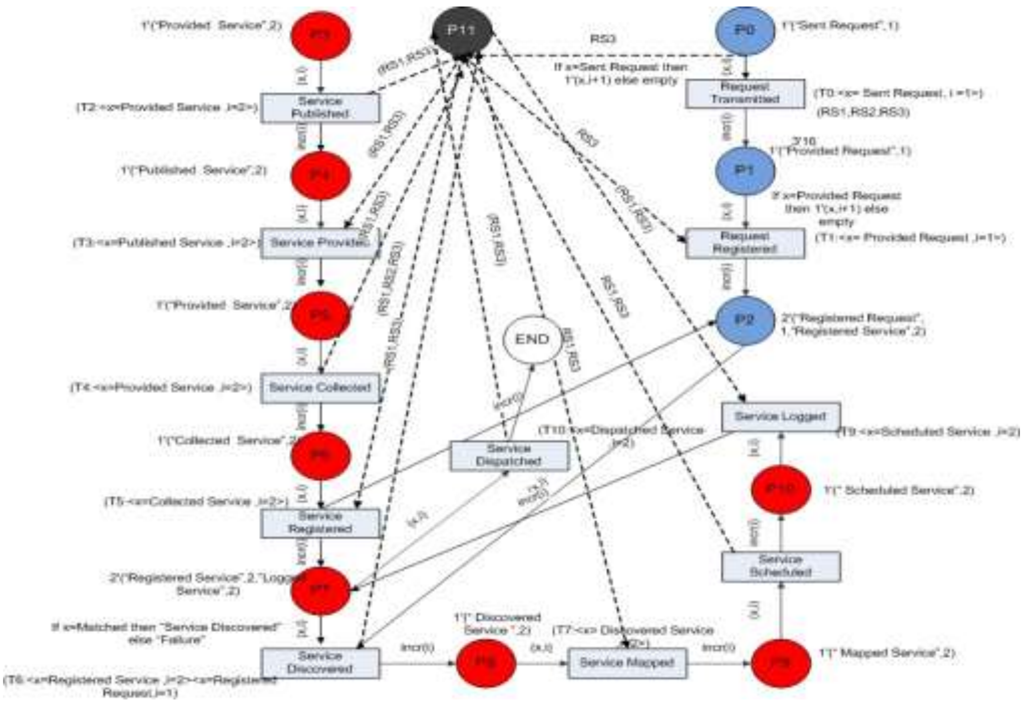


Figure 2. High Level Enterprise Cloud Bus Petri Net

The process starts from a place *P0* which is the client and after a transition *T0* will reach a place *P10* from which the scheduling agent of place *P7* will collect the service for delivering it to the client.

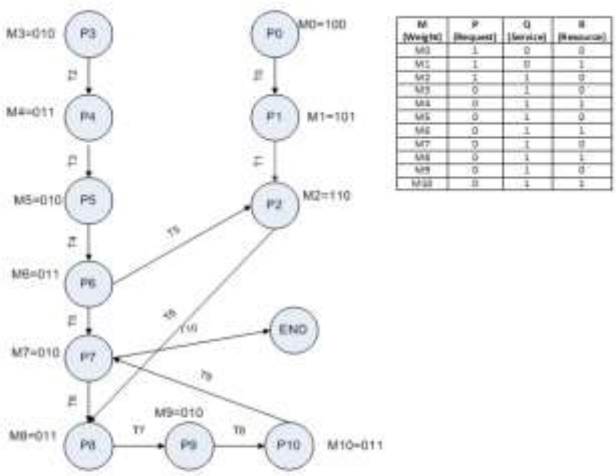
The process continues further on and we finally arrive at the place *P7*. Serially as the transitions occur the process moves on to each of the places as explained in the tables. The place *P11* is the places for the resources *RS1*, *RS2* and *RS3* respectively.

All the cloud bus components will request each of the resources as and when required and once the transition is fired will release it updating the knowledge base. In this system a place have token such that, *Token* $\rightarrow C \times K \times PR \times S \times R \times I$. From the *HECBP* net, the corresponding Reachability Graph is obtained and shown in Figure 3.

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

(a) Reachability: Reachability property is a fundamental artifact for analyzing the dynamic properties of any *MAS* based cloud architecture. This properties states whether a marking *M'* is reachable from an initial marking *M* i.e., whether there exists a finite number of occurrence sequence from *M* to *M'*.

However in this section it has been established from figure 3 that all the markings in the *HECBP* net are reachable starting from any marking in the net and hence reachability exists. This guarantee that the *HECBP* net modeled the *ECBS* that will meet the pre-specified goal;



M (Weight)	P (Request)	Q (Service)	R (Resource)
M0	1	0	0
M1	1	0	1
M2	1	1	0
M3	0	1	0
M4	0	1	1
M5	0	1	0
M6	0	1	1
M7	0	1	0
M8	0	1	1
M9	0	1	0
M10	0	1	1

Figure 3. Reachability Graph Corresponding to Figure 2

(b) Home Properties: A marking in the *HECBP* is said to be a home marking if it is a home space. It tells us about the markings where it is possible to return back. In the proposed *HECBP* net, *M0* is the initial marking which is considered to be a home marking that and M0 ∈ M and a set of markings $Z \subseteq M$ be given as:

M0 is a home marking, *if:* $\forall M' \in [M0]>: M \in [M'];$ and *Z* is a home space *if:* $\forall M' \in [M0]>: Z \cap [M']> \neq \emptyset]$. In this context, *M* is a home marking if {*M*} is a home space.

(c) Boundedness: The boundedness property states that after considering all reachable markings, the number and type of tokens a place may hold in the net. It can be concluded after analyzing the *HECBP* net that there is no unboundedness at any stage once the process starts and goes from place *P0* to *P7* via *P10* and thus the boundedness of the *HECBP* net is guaranteed and safe;

(d) Liveness: The liveness properties of a *HECBP* model shows a continuous dynamic operation of the proposed net model and ensure that the system is live once transitions are fired. In the proposed *HECBP* net as the component of the cloud bus process starts from *P0* transitions *T0* through *T10* are fired and place *P7* is reached. The net is continuous and hence the liveness property is ensured. Thus the proposed net is live;

(e) Fairness: The net HECBP is said to be bounded-fairness because at a time single transition are fired. It can also be termed as unconditional fairness because every transition appears infinitely in a firing sequence. Here the net is B-Fair.

(f) Safeness: Any place in a *HECBP* net is considered as safe if number of tokens at that place is either 0 or 1 and there is no deadlock present in the net. Therefore, the concerned net as a whole is declared safe.

5. Simulation of HECBP

There are various tools to analyze Petri Net based System behavior. In this section, *HECBP* Net is analyzed using CPN tools to study the behavioral aspects of Multi-Cloud architecture defined using proposed conceptual model of *ECBS*.

Here, three colors sets namely red, blue and black have been used to depict the three types of tokens namely request, service and resource token respectively. Thus few restrictions have been imposed for simulation of the *HECBP* Net using CPN simulation and are summarized as follows,

(a) Before and after Transitions the data types of Tokens have to be same or else transition will not be fired / enabled;

(b) During the simulation process, at any point of time, it cannot be clearly expressed which component is enabled after a Transition is fired;

(c) Multi Token pass can be done but only one single token is being passed at a time.

The advantage of using the concept of High level Petri Net along with CPN simulator tool for analyzing dynamism of multi- cloud behavior is as follows:

(a) The dynamic component (*PA, CA, SA*) can be handled with ease.

(b) Validation of the agent based Multi- cloud architecture can be done.

The declarations for the generated *HECBP* net corresponding to the Figure 2 can be expressed as,

```
colset BOOL=bool;
colset INTINF = intinf;
colset TIME = time;
colset REAL= real;
colset UNIT= unit timed;
colset STR= with S timed;
var x: STR;
colset IN = with P | Q | R timed;
var i: IN;
colset PRO= product STR * IN timed;
var p: PRO;
```

5.1. HECBP Simulation through CPN Tool

In this section, the *HECBP* net is simulated using CPN tool and the corresponding simulation results before and after transitions are shown. Figure 4 shows the simulation before resource is allocated to the place *P0*. Once the requested resources are allocated to the place *P0* the relevant transition *T0* is fired and place *P1* is reached. The resources are held up by the place *P1*. They will be returned to the resource pool and only then the next transition will be enabled.

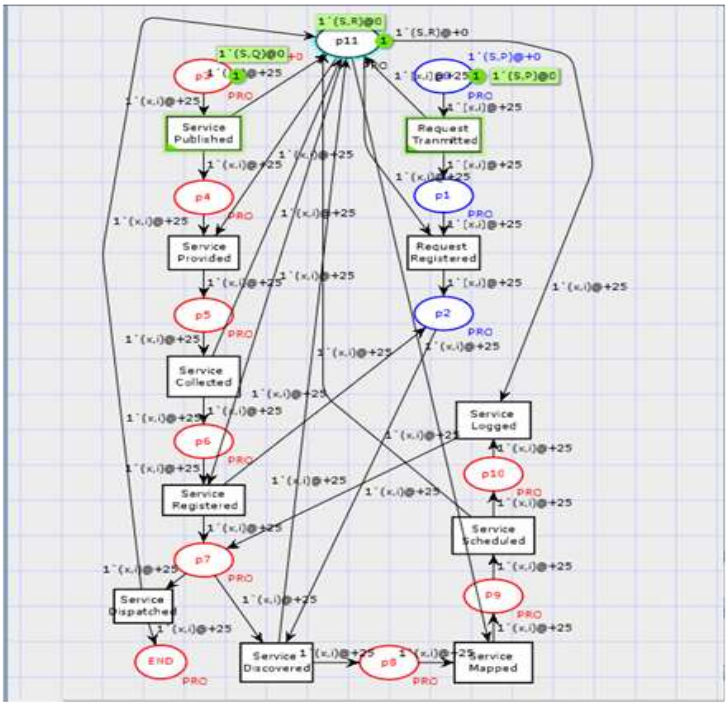


Figure 4. CPN simulation - Before Resource Allocation

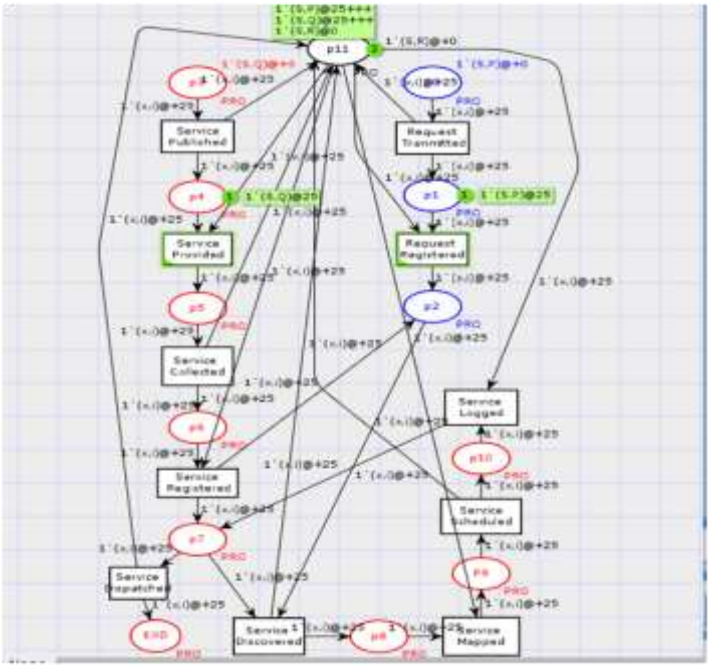


Figure 5. CPN simulation - After Resource Allocation

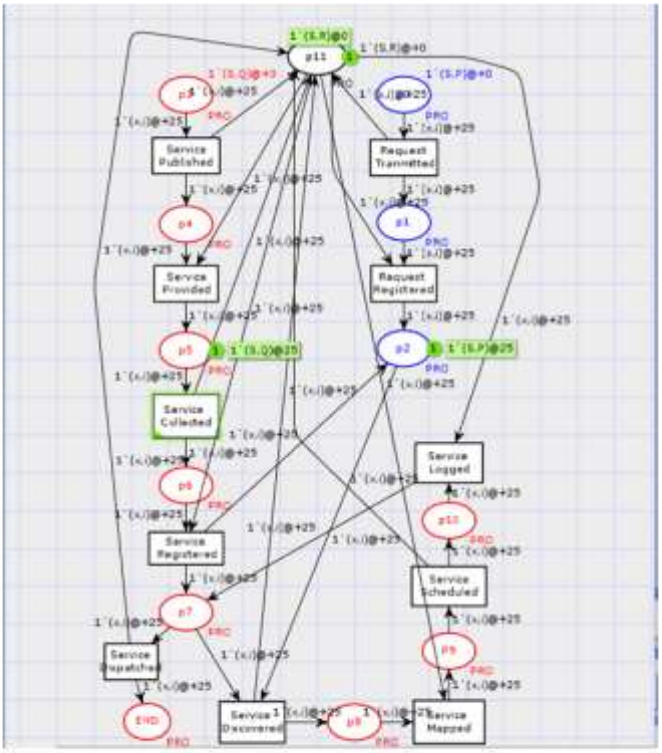


Figure 6. CPN simulation - Resource used and released

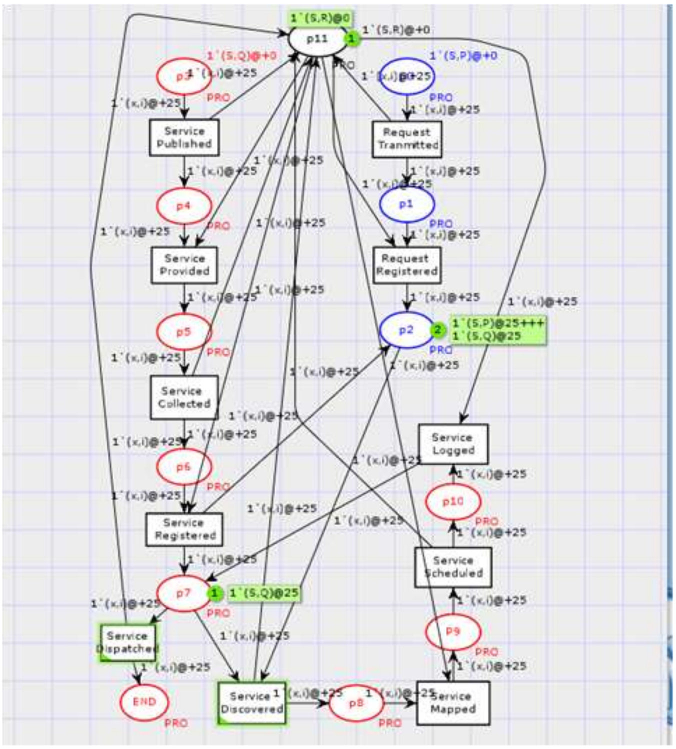


Figure 7. CPN simulation - Service discovered by SA

Figure 5 shows the simulation after resource is allocated to the required place. Once these resources are released, as shown in Figure 6, next transition *T1* will be enabled and place *P2* will be reached. Hence it can be seen that the resources are allocated and released dynamically. It is this dynamicity of the cloud bus component properties that is very smoothly analyzed with the help of the proposed model.

Further in Figure 7 the services are discovered once the request token in *CUDDI* is matched with the service token by the scheduling agent. Finally, services are dispatched and resources are released.

6. State Space Analysis of HECBP

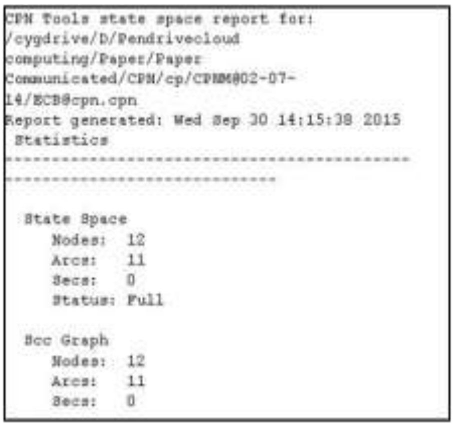
Simulation performs the analysis of the model in a finite number of sequences. On conducting state space analysis of the model, the simulation tool usually generates report based on its state space that gives the detail study of the generated state space and the behavioral characteristics of the proposed net.

The report also gives a clear idea about the beat upper and lower bounds. After simulation of the *HECBP* net, the corresponding state space reports are shown. The state space report (Figure 8) gives the state space statistics. The boundedness properties are shown in Figures 9 that tells the number of tokens in a place after considering all reachable markings. The best upper and lower bounds results are shown in Figure 10.These reports exhibit that the proposed *HECBP* is bounded and safe.

Figure 11, specifies the home properties, liveness properties and fairness properties. In Home properties the home marking is node *P12*. Even in liveness property node *P12* is regarded as dead because the execution of the process ends at that node namely *P12* from which scheduling agent (*P7*) takes the services and dispatch it to the end users. It is observed that the *HECBP* net is live because there are no dead transitions.

Similarly, we proved that the fairness property also is true since there is no infinite occurrence sequence in the net. In the theoretical analysis a discussion was made on *ECBS* and further using formalized and conceptual definition of the proposed *ECBS* a corresponding *HECBP* model and its reachability graph was obtained and through which basic dynamic behavioral properties like liveness, safeness, and boundedness were proved.

It was observed and established from the theoretical analysis that all the dynamic properties of *ECBS* were true for the *ECBS* considered as discussion. CPN tool was used to successfully simulate and design the *HECBP* model. Once the *HECBP* is simulated successfully, the State Space Analysis generates the State Space Reports which shows that all the dynamic behavioral properties of the *HECBP* are successfully demonstrated and verified. Thus the simulation results of the *HECBP* model strongly validate the theoretical analysis.

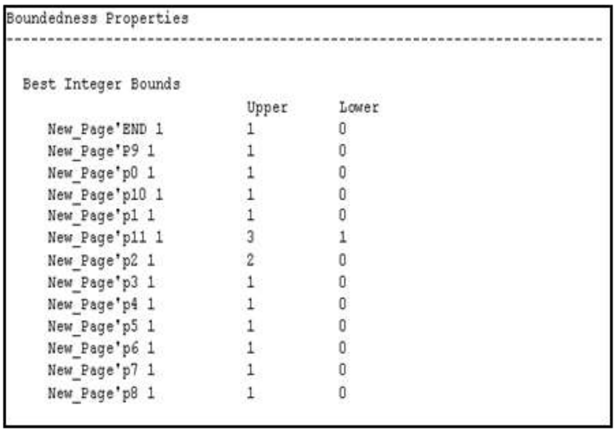


```
CPN Tools state space report for:
/cygdrive/D/Pendrivecloud
computing/Paper/Paper
Communicated/CPN/cp/CPNM@02-07-
14/ECB@cpn.cpn
Report generated: Wed Sep 30 14:15:38 2015
 Statistics
------------------------------------------
---------------------------

  State Space
     Nodes:  12
     Arcs:   11
     Secs:   0
     Status: Full

  Scc Graph
     Nodes:  12
     Arcs:   11
     Secs:   0
```

Figure 8. State Space Report



```
Boundedness Properties
------------------------------------------------------------------

 Best Integer Bounds
                         Upper      Lower
    New_Page'END 1       1          0
    New_Page'P9 1        1          0
    New_Page'p0 1        1          0
    New_Page'p10 1       1          0
    New_Page'p1 1        1          0
    New_Page'p11 1       3          1
    New_Page'p2 1        2          0
    New_Page'p3 1        1          0
    New_Page'p4 1        1          0
    New_Page'p5 1        1          0
    New_Page'p6 1        1          0
    New_Page'p7 1        1          0
    New_Page'p8 1        1          0
```

Figure 9. Boundedness Properties


```
Best Upper Multi-set Bounds
     New_Page'END 1        1`(S,Q)
     New_Page'P9 1         1`(S,Q)
     New_Page'p0 1         1`(S,P)
     New_Page'p10 1        1`(S,Q)
     New_Page'p1 1         1`(S,P)
     New_Page'p11 1        1`(S,P)++
1`(S,Q)++
1`(S,R)
     New_Page'p2 1         1`(S,P)++
1`(S,Q)
     New_Page'p3 1         1`(S,Q)
     New_Page'p4 1         1`(S,Q)
     New_Page'p5 1         1`(S,Q)
     New_Page'p6 1         1`(S,Q)
     New_Page'p7 1         1`(S,Q)
     New_Page'p8 1         1`(S,Q)

  Best Lower Multi-set Bounds
     New_Page'END 1        empty
     New_Page'P9 1         empty
     New_Page'p0 1         empty
     New_Page'p10 1        empty
     New_Page'p1 1         empty
     New_Page'p11 1        1`(S,R)
     New_Page'p2 1         empty
     New_Page'p3 1         empty
     New_Page'p4 1         empty
     New_Page'p5 1         empty
     New_Page'p6 1         empty
     New_Page'p7 1         empty
     New_Page'p8 1         empty
```

Figure 10. Upper and Lower Bounds

```
Home Properties
------------------------------------------------------------------------
  Home Markings
     [12]
Liveness Properties
------------------------------------------------------------------------
  Dead Markings
     [12]
  Dead Transition Instances
     None

  Live Transition Instances
     None

Fairness Properties
------------------------------------------------------------------------
     No infinite occurrence sequences.
```

Figure 11. State Space Report of other behavioral Properties

7. Conclusion and Future Work

Multi-agent based Inter-cloud architecture represents dynamic and complex system that consists of heterogeneous autonomous entities (*CA*, *PA*, *SA*) and other components present inside the cloud bus. These autonomous entities play some specific roles in the system. Based on these roles, collaborations occur between the participating agents and components in *ECBS*. Participating agents in *ECBS* are proactive and thus interact with the Multi-cloud environment or with some other components in the system. Thus, collaborations and interactions among the participating agents and components are the prime factors to design the dynamics of ECBS effectively. As a result of this dynamicity of ECBS, there are various behavioral properties exist in the *ECBS*. For the purpose, a High Level Enterprise Cloud Bus Petri Net (*HECBP*) has been established in this paper to analyze and model the behavioral aspects of *ECBS* based multi-cloud architecture.

The dynamicity of the proposed enterprise cloud bus system benefits the enterprise applications in optimizing the performance, cost, elasticity, flexibility, high reliability and availability of the computing resource. Further, a set of mapping rules have been described for representing the elements of the proposed conceptual framework of the

ECBS into the *HECBP* net. The key benefits of using the proposed mechanism are the capability to represent study and analyze the interactions among the multiple agents present inside the Cloud Bus. Using the proposed *HECBP* concepts and corresponding reachability graph, the behavioral properties of an *ECBS* like reachability, safeness, Boundedness, liveness can be analyzed formally. Moreover, simulation of the proposed *HECBP* with CPN tool and generated results strongly validate the proposed claims.

Future work includes the quality analysis of *ECBS* dynamics from the proposed concepts of *HECBP*. Development of a dedicated simulation tool for the conceptual architecture of *ECBS* and *HECBP* is also a prime future objective

References

[1] Z. Zheng and M. R. Lyu, "Collaborative reliability prediction of service-oriented systems", In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, vol. 1, **(2010)**, pp. 35-44.

[2] P. Banerjee, A. Sarkar and Narayan C. Debnath, "Analytical Model for Component Based Software: Towards Effective Design", International Journal of Software Engineering and Its Applications, vol. 9, no. 12, (**2015)**, pp.13-40.

[3] A. Mandal, S. Changder, A. Sarkar and N. Debnath, "Architecting Software as a Service for Data Centric Cloud Applications", International Journal of Grid and High Performance Computing (IJGHPC), IGI-Global, USA, vol. 6, no. 1, **(2014)**, pp. 77 – 92.

[4] E. Cavalcante, "Architecture Driven Engineering of Cloud-Based Applications", IFIP/Springer-Verlag, Germany, vol. 22, **(2013)**, pp. 175-180.

[5] K. Alexandros, G. Aggelos, S. Vassilis, K. Lampros, P. Magdalinos, E. Antoniou, Z. Politopoulou, "A cloud-based Farm Management System: Architecture and implementation", In Journal Computers and Electronics in Agriculture, vol. 100, **(2014)**, pp.168-179.

[6] R. Buyya, R. Ranjan, N. Rodrigo, "Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services", In Algorithms and architectures for parallel processing, LNCS Springer, vol. 6081, **(2010)**, pp. 13-31.

[7] D. Ardagna, E.D. Nitto, E. Casale, P. S. Mohagheghi, P. Mosser, A. Matthews, C. Gericke, F. D. Balligny, C.S. Nechifor, C. Sheridan, "MODACLOUDS, A Model-Driven Approach for the Design and Execution of Applications on Multiple Clouds", International Workshop on Modelling in Software Engineering, **(2012)** pp. 50–56.

[8] N. Ferry, R. Alessandro, C. Franck, M. Brice and Arnor Solberg, "Towards model-driven provisioning, deployment, monitoring, and adaptation of multi-cloud systems", In 2013 IEEE Sixth International Conference on cloud computing, **(2013)**, pp. 887-894.

[9] E. Elmroth, J. Tordsson, F. Hernández, A. Ali-Eldin and R. Pette, "Self-management challenges for multi-cloud architectures, Lecture Notes in Computer Science Towards a Service-Based Internet", vol. 6994, **(2011)**, pp. 38-49.

[10] A. Sarkar, "Modeling multi-agent system dynamics: Graph semantic based approach", 10th International Conference on Service Systems and Service Management, **(2013)**, pp. 664-669.

[11] P. Dhavachelvan, S. Saravanan and K. Satheskumar, "Validation of Complexity Metrics of Agent-Based Systems Using Weyuker's Axioms", International Conference on Information Technology (ICIT '08), **(2008)**, pp. 248- 251, 2008.

[12] A. Sarkar and N. C. Debnath, "Measuring Complexity of Multi- Agent System Architecture", 10th IEEE Conference on Industrial Informatics, **(2012)**, pp. 998 – 1003.

[13] B. Djamel, "An agent-based approach for hybrid multi-cloud applications", Journal of Scalable Computing: Practice and Experience, vol. 14, no. 2. **(2013)**, pp. 95- 109.

[14] G. Khan, S. Sengupta, A. Sarkar and N.C. Debnath, "Modeling of Inter-Cloud Architecture using UML 2.0: Multi-Agent Abstraction based Approach", 23rd International Conference on Software Engineering and Data Engineering, **(2014)**, pp. 149-154.

[15] G. Khan, S. Sengupta, A. Sarkar and N.C. Debnath, "Web Service Discovery in Enterprise Cloud Bus Framework: T Vector Based Model", 13th IEEE International Conference on Industrial Informatics (INDIN 2015), UK, **(2015)**, pp. 1672 – 1677.

[16] G. Khan, S. Sengupta, A. Sarkar and N.C. Debnath, "Performance Analysis of Service Discovery for Enterprise Cloud Bus (ECB)", IEEE International Conference on Industrial Technology (ICIT 2016), **(2016)**, pp. 1728-1735.

[17] G. Khan, S. Sengupta and A. Sarkar, "Priority based Service Scheduling in Enterprise Cloud Bus Architecture", IEEE International Conference on Foundations and Frontiers in Computer, Communication and Electrical Engineering (C2E2 2016), **(2016)**, pp. 363-368.

[18] F. Ahmad and A. Sarkar, "Scheduling of Composite Services in Multi-Cloud Environment", 11th International Conference on Grid & Cloud Computing and Applications (GCA 2015), WorldComp 2015, **(2015)**, pp. 10 – 15.

[19] B. Marzougui, K. Hassine, and K. Barkaoui, “A New Formalism for Modeling a Multi Agent Systems: Agent Petri Nets”, Journal of Software Engineering and Applications, vol. 3, no. 12, **(2010)**, pp 1118-1124.
[20] S. Pujari and S. Mukhopadhyay, “Petri Net: A Tool for Modeling and Analyze Multi-agent Oriented Systems”, Intl. Jnl. Intelligent Sys. and Appls., vol. 10, **(2012)**, pp.103-112.
[21] R. K. Chatterjee, A. Sarkar and S. Bhattacharya, “Modeling and Analysis of Agent Oriented System: Petri-net Based Approach”, 11th Intl. Conf. on Software Engineering Research and Practice (SERP 11), vol. 1, **(2011)**, pp. 17 – 23.
[22] B. Sofiane, H. Bendoukha and H. Moldt, “ICNETS: Towards Designing Inter-Cloud Workflow Management Systems by Petri Nets”, In Enterprise and Organizational Modeling and Simulation, 198. Springer International Publishing, **(2015)**, pp. 187- 198.
[23] G. Khan, S. Sengupta and A. Sarkar, “Modeling and Analysis of Enterprise Cloud Bus using a Petri Net Based Approach”, 3rd International Doctoral Symposium on Applied Computation and Security Systems (ACSS) , Springer, **(2016)**, pp. 17-36.
[24] Z. Jun, W. H. Ngan, L. Junfeng, W. Jie and Y. Xiaoming, “Colored Petri Nets Modeling of Multi-agent System for Energy Management in Distributed Renewable Energy Generation System”, Asia-Pacific Power and Energy Engineering Conference (APPEEC), **(2010)**, pp.28-31.
[25] R. Chatterjee, Neha and A. Sarkar, “Behavioral Modelling of Multi Agent System: High Level Petri Net Based Approach”, International Journal of Agent Technologies and Systems, vol. 7, no. 1, **(2015)**, pp. 55 - 78.