Controlled Stochastic Petri Net Model for End-to-End Network QoS Provisioning in Middleware-based Multimedia and Real-Time Systems

Hakiri Akram[1,2], Berthou Pascal[1,2], Gayraud Thierry[1,2]
[1]CNRS ; LAAS ; 7 avenue du colonel Roche, F-31077 Toulouse, France
[2] Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse, France
{Hakiri, Berthou, Gayraud}@laas.fr

Keywords: Controlled SPN, Quality of Service, Multimedia System, Heterogeneous Networks

Abstract

End-to-end quality of service (QoS) is central to the objectives of the today's networks requirements of middleware based distributed real-time and embedded (DRE) systems. Any middleware based QoS system should be totally oriented to this goal, and in the scope of this purpose several mechanisms, components and approaches were, are being and will be developed in order to achieve it. In this paper, we show how controlled behavior of such QoS-aware systems can be developed based on stochastic Petri Nets. Afterwards, We show how to obtain, using such an interpreted formal model, powerful numerical analysis for the management of the network QoS.

1. INTRODUCTION

Distributed real-time and embedded (DRE) middleware-based systems refer to a distributed platform of interfaces and services that reside 'between' the application and the operating system and aim to facilitate the development, deployment and management of distributed applications. They often comprise multiple end-to-end application flows that may require various QoS properties affecting CPU, memory, and network resources[11]. Examples include, among others, unmanned air vehicle, experimental aircraft for disaster recovery, data-centers, space mission systems, video surveillance, real-time and on demand video transmission, homeland security, on line stock trading, and weather monitoring. In most of these systems, the right answer delivered too late becomes the wrong answer, i.e achieving the end-to-end QoS is essential. Most challenging DRE systems will operate in large scale, distributed and heterogeneous environment that take data from large number of sensors, multimedia sources, real-time streaming systems, and remote control systems [10].

To address the run-time QoS demands, research efforts focus on the QoS-enabled components and emerging technologies including middleware; the basic mechanisms for end-to-end QoS provision are located at the data plane (e.g. packet scheduling) and control plane (e.g., admission control and resources reservation). Signalling is carried out at various levels: end-to-end, at application level, through the use of SIP [1]; hop-by-hop, at inter-domain level, complemented by routing and/or traffic engineering; locally, at intra-domain level, for resource management and provision. Likewise, the middleware is a distributed software layer which shields the application from the complexity and heterogeneity of the underlying distributed environment. The middleware was designed to help the development and management of distributed systems. Examples of middleware include the High Level Architecture (HLA), Java Message Service (JMS), CORBA Event Notification, OMG Data Distribution Service (DDS), and Grid Computing.

Recently, significant research has been made towards making software as industrial reusable components, evolving access to information, computer and network resources over middleware-based distributed systems. In order to assess the adequacy of our middleware model-based approach to support effective QoS policies, our approach consists of the following:(1) identification of the QoS policies and the analysis of the QoS constraints, (2) proposal of Stochastic Petri Net model-based approach for middleware-based Network QoS provisioning. To the best of our knowledge there is not any other published paper dealing with this specific topic of QoS provisioning including together the application, the middleware and the network resource allocation. In this paper, we show how controlled behavior of such QoS-aware systems can be developed on the control plane and how it invokes the middleware services using some kind of Petri Net formalism.

The remainder of this paper is organized as follows: after this introduction, Section 2 briefly describes the theory of Generalized Stochastic Petri Nets (GSPN). Section 3 uses case studies to motivate common requirements of middleware-based multimedia and DRE systems associated with the network QoS provisioning. Section 4 analysis the QoS policies that should be managed for such system and addresses the key challenges associated with meeting these requirements. GSPN middleware-based model used in our approach is described in section 5, as well as the results from simulation. Section 6 compares our work with related researches; finally, some concluding remarks and future works are presented in Section 7.

2. GENERALIZED STOCHASTIC PETRI NETS

In most QoS research area, model driven approach [14], state machine [13], dataflow analysis [15] are used to explore possible solution for QoS assurance between components; they focus on deterministic process analysis and how QoS parameters can interoperate between each other. However, these approaches do not suffice for QoS analysis, because when analyzing the QoS provisioning, additional information's are needed: when an application is sharing differ-

ent flows, each flow may have specific QoS parameters. For example, fire sensor should not have the same data priority as camera surveillance; data should be sent in reliable transport service with high priority. Hence, providing some kind of urgency, probability of events occurring and precedence between camera flow and fire sensor flow is very important. All these characteristics show the difficulty for the previous approaches to ensure the QoS requirements of DRE systems.

Generalized Stochastic Petri Nets (GSPN) [2] are well-known interpreted extensions of autonomous Petri Net (PN) [2]. Petri Nets are similar to those approaches but have additional benefits in modeling. Since graphs in Petri Nets allow the representation of possible assembled DRE systems, it consider the performance aspect in the design of distributed and complex concurrent systems. Probabilistically time delay is introduced in transitions to allow performance evaluation and resolving scheduling problems in DRE. In addition to the PN's capability of structural and functional validation, GSPN provides numerical solution for efficient algorithms with Markov Chain (MC) [2] to get the exact performance parameters such as throughput and delay.

Definition

GSPN=(P, T, I, O, H, M_0,Π,W) is the Petri Net comprising the following:

P is a set of places and T is a set of Transitions such that P $\cap$ T= $\oslash$.
Let introduce a (possibly empty) set T' $\subset$ T of immediate transitions. Transitions in T-T' are called timed transition.

I(t), O(t), H(t): T$\times$P$\longrightarrow \mathbb{N}$, are the input, output and inhibitor function, respectively, for each transition t $\in$ T.

M_0:P $\longrightarrow \mathbb{N}_+$, is the initial marking: a function that assigns a non negative integer to each place

Π:P$\longrightarrow\mathbb{N}$, is the priority function which associates lowest priority (0) with timed transitions and higher priorities (≥ 1) with immediate transitions:

$$\Pi(t) = \begin{cases} 0, & \text{if t is timed} \\ \geq 1, & \text{if t is immediate} \end{cases} \quad (1)$$

W:T$\longrightarrow \mathbb{R}$ is the weighting function that associates any transition with a real value (it would be great if you can explain what it means). Besides, note that you should use $\omega(t)$ instead of W.

- the parameter of the negative exponential probability density function (pdf) of the transition firing delay, if t is a timed transition,
- a weight used for the computation of firing probabilities of immediate transitions, if t is an immediate transition.

Simulation: An XML based standard discrete-event simulation has been implemented for the performance evaluation of GSPN models. It allows arbitrary distributions of firing delays including zero delays, complex token types, global guards, time guards, and marking dependent transition priorities.
Petri net classes are defined by an extendable XML schema in TimeNET 4.0 [20] which affects the behavior of the graphical user interface. TimeNET has numerous merits: flexibility, extensibility of the model, clear XML syntax, user friendly graphical interface to show the simulation in run-time, debugging, and a scripting engine allowing script-controlled generation and modification of the models as well as an automated start of the simulation with different parameters.

3. MOTIVATING MIDDLEWARE-BASED NETWORK QOS PROVISIONING

Before detailing the GSPN model, a brief example is given to illustrate how and why the QoS GSPN-based model allows the design, specification and evaluation of QoS requirements of DRE systems.

3.1. Challenge 1: Alleviating the ambiguity in QoS Specification

Context. Network architecture having more complexity and quality requirements will need more precise specification for better control and design [11]. Cooperative enterprizes often use IP network over high speed network. Traffic can be grouped into several classes, including email, video conference, fire/smoke sensors data, camera surveillance, imagery traffic, etc. To provide transmitting the right data at the right place in the right time, each flow must specify its requirements to the underlying transport network service. As pointed out in [12], DRE systems can be specified, modeled, analyzed, and evaluated with the aid of the structural and behavioral analysis of Petri Nets formalism.

Problem. When constructing DRE systems, developers need to make decisions about the cooperative aspect of different DRE components. Each decision may induce diverse execution orders, execution time, and events. Therefore, there are huge number of possible of cases generated bases on different decisions [12]. To reduce the complexity of exploring all the possible cases by evaluating their QoS requirements, the evaluation should expresses the specific time before/after the events affect the QoS parameters, the kind of parameters that should be evaluated, and how to carry out the relationship between them. Hence, there are no mature approach, principle or solution to enable large scale DRE systems to be QoS predictable, repeatable, validated, and enhanced.

Solution Approach. In our approach, we show that the modeling begins with a representation of adequate mechanisms for each application, and is, after each step, followed by the validation. Then, different behavior can be represented and their global behavior evaluated. The specification will be considered correct, when, after validation the general properties of the model are correct.

3.2. Challenge 2. Alleviating the complexity of resource reservation and deployment

Context. The manner on which the reservation strategies in presence of middleware depends on the number and the type of calls received from different hosts [16]. For example, Homeland security sensors and video surveillance camera are more sensible for delays, bandwidth. Basically, each kind of traffic should specify its QoS requirements [17] to the network during the call admission control phase with the aid of some kind of signaling protocol.

Problem. This area represents the next great wave of evolution for middleware. Despite the ease connectivity provided by middleware, however, the DRE traffic has always commanded to respect some kind of Service Level Agreement (SLA) [4] to avoid the situations where penalties may be paid. The premises underlying the push towards end-to-end resources reservation mediated by middleware are that different level of service are possible under different conditions, costs, and constraints. A tradeoff between the levels of service is used to respect the constraints.

Solution Approach. The service level involved between the middleware and the control plane is fulfilled, when, after validation of its properties, the following problems are solved: accessibility (how to grant and control the access to the service), controllability (how to control service instances), and resources allocation (how to locate and define the admission control strategy and the resources reservation). The model developed in this approach provides one way to translate the QoS requirements from one level to another (application, middleware, transport, and network). For example, as traffic from video application is expected to be a substantial portion of the traffic carried by emerging wired and wireless networks, video source should use variable bit rate at the encoder to manage the errors, packet lost (especially in wireless networks) and delay (delay is annoying to the viewer) at the application level and notify the middleware about these changes. Middleware should, at its turn, notify the control plane to adapt the resource reservation strategy to changes.

3.3. Challenge 3. Reducing the complexity of the multi-domain end-to-end QoS guarantee

Context. To support end-to-end QoS for DRE applications, the complete chain of involved systems and the network should support it. Ideally, applications make use of end-to-end signaling to signal its end-to-end QoS requirements and agreed QoS between each other [18]. For instance, Video on demand applications should include traffic description (delay, bandwidth, jitter, packet lost,...) describing which kind of treatments by the network element. Since the Differentiated Service (DiffServ) and the Integrated Service (IntServ) [4] are the most used architecture for end-to-end QoS provisioning, flows having different priorities, delays, and bandwidth requirements, have to obtain the required network resources fulfilling these requirements. The network backbone (access and core network) should be able to allow the transmission of each flow without affecting other flows, and permit the call admission control of incoming demands.

Problem. Despite the advances of the currently available network architecture and models for resources handling (DiffServ, IntServ), however, it is not sufficient to deploy them within routers to guarantee QoS between two entities in the Internet. End-to-End QoS system should be able to coordinate the routers in the network according to the users' demands. For example, communication provided the nowadays Internet infrastructure has, by nature, multi-domains and multi-technologies architecture. Data packet travel across many domains, including diverse Internet Service Providers (ISP) in different countries, each has its own deployed network backbone, what makes hard to provide end-to-end QoS guarantee [11].

Solution Approach. The concept of policy control developed in our approach aims to provide QoS SIP-based [1] Petri Nets model to allow application expressing their QoS requirements using text-based exchanged messages. Like shown in Figure 1, the SIP/SDP [19] model coordinates the application signaling, the resources reservation and provides to the Bandwidth Broker (BB) [4] the facilities to coordinate the routers in the network, that applies to both IntServ and DiffServ infrastructure. Under the proposed Petri Net model we consider an admission control policy fulfilling both per-flow and Class-based end-to-end guaranteed service to allow QoS-enabled network. Stochastic priorities assigned to transitions in GSPN model allow the better scheduling of different incoming calls to satisfy the QoS requirements of each flow. DRE applications can select the necessary firing time depending on the probabilistic value of the timed transition.

4. GLOBAL MODEL DESCRIPTION

It is important to define a common modeling system, using a level of abstraction that will permit the quantitative analysis of non-trivial system, then we will detail the architecture of the proposed model including the application, the middleware and the network. Figure 1 describes the global architecture of

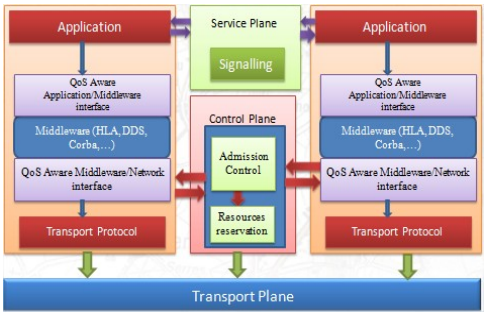

Figure 1. Global system architecture

the system based on the following entities:

Application: A multimedia application for Unmanned Air Vehicle (UAV) video distribution using multi-layer resource management mechanisms are coordinated via middleware to ensure video flows can meet their mission QoS requirements by adapting both the computer and the network resources. The

architecture adaptively control the video transmission captured from cameras, then send them to middleware-based distributed process to be distributed over QoS-Aware network to one or more remote receivers (end-systems) including displayers and image processing software.

Application/Middleware Interface: it aims providing users the ability to specify high-level management policies: the registration mechanisms and controlling how flows are exchanged and delivered.

Middleware: when considered in the GSPN Model, it is considered as a Message-Oriented Middleware (MOM). Moreover, the middleware is composed of Message Manager and Queue Manager to process incoming flow. This abstraction facilitates the modeling of the whole system.

Middleware/Network Interface: it uses a QoS allocator to notify the Control Plane about the middleware data and the messages that will be exchanged during the reserved section.

Service Plane: Application will use SIP to reserve the communication path and channel signaling. For example, Multimedia application use it to the renegotiate the codec, but it uses classical transport protocol for data transmission.

Control Plane: This common control plane promises to simplify network operation and management by automating end-to-end provisioning of connections, managing network resources and providing the level of QoS. It includes two mechanisms: the admission control unit and the resource reservation. The transport plane is an abstraction of the underlying interconnection technology.

5. GSPN MODEL FOR QOS PROVISIONING

The architecture of the GSPN model for QoS provisioning is presented in Figure 2. We assume, for clarification seeks, that we have 2 types of flow, but the model can be easily extended as well as for great number of streams.

5.1. Modeling

The main components are the Application (App), the Service Plane (SIP), The Middleware Interface (MI), the middleware (Midd), and the Control Plane (CP):

The application: is composed of the sender application (AppS) and the receiver application (AppR), and it is responsible for sending and receiving packets to and from the middleware, respectively. Those packets represented by tokens in the places Sender and receiver to show that application is ready to send/receive data. When modeling the packet lost of multimedia stream, the permissible delay between encoding and playback may be too low to allow retransmission or channel coding. A practical and widely used approach to this problem is considering that the codec can process the Packet-Loss Concealment (PLC) [5] schemes and masks the loss of a packet by using information from the last good packet. Hence, in the sender side, the codec self correction mechanism is represented in transition T1 by the stochastic arrival process with 'pdf' distribution "λ_{codec}". Since most modern decoders are capable of performing error concealment to mitigate losses, feedback can be used to notify the encoder of the losses, in particular applications using Session Description Protocol (SDP) need to agree on what voice codec to use for transmission and reception, the place "codec_buffer" represent the way to provide a loss-aware rate-distortion scheme that can be used by both SIP and SDP protocols [1].

The Service Plane (SIP): In this model we consider that we have two User agents at both the sender side and the receiver side described by the places *'SIP UA1'* and *'SIP UA2'*, respectively. Each Agent is capable of generating requests or sending a response to a request. It is the user agents that react to one another. When a dialog is created, it is a user agent communicating with another user agent. The challenge 1 can be verified, when, looking at Table 1 and Table 2, the semantics of the SIP bloc using PN formalism, describing the behavioral and structural proprieties of this model is bounded, deadlock free and liveliness. We check whether or not a UA1 will wait infinitely. The net does not fulfill this property: starting from the initial marking, We can reach this marking at any time of the simulation (The reachability graph is not represented here, but it can be found intuitively). The model is well suitable for end-2-end signaling between the end applications. More, the effect of the firing sequence is determined by the incidence matrix and the vector of transition sequence, and has an effect on the behavior of the model. Since the number of marked places at the same time is equal to 1 at both sides of the SIP bloc, the protocol is conservative and only one operation is permitted at any given time.

Table 1. Specification of the places in SIP bloc of Figure 2

Place 1	Semantics
PSIP1	Wait for UA2 rcv replay
PSIP5	Wait for UA2 rcv replay
PSIP2	Wait for ACK from UA2
PSIP6	Wait for ACK from UA1
PSIP3	Wait for Session connection with UA2
PSIP7	Wait for Session connection with UA2
PSIP4	Wait for Termination with UA2
PSIP8	Wait for termination with UA1
PSIP9	Invite() initialized from UA1
PSIP10	Ringing() replay from UA2
PSIP11	OK response from UA2
PSIP12 PSIP13	ACK response from UA1, session initiated, BYE message send from UA2, it decides to end the session
PSIP14	200 OK, the replay from UA1 to agree the session end

Middleware Interface: in addition to the Concealment scheme provided by the coded, the middleware provides, via place P3, a notification interface to adapt the application bit rate to allow better user quality of experience. Further, when a token is present in place P3, this means that the network states is not able to transmit all the data incoming from the application. This case can occur in Internet network, where the number of client being admitted is greater. Hence, the quality of

Table 2. Specification of the transitions in SIP bloc

Transitions 1	Semantics 2
T221	UA1 sends connection Request to UA2
T22	UA2 receive the Request
T222	UA2 sends Replay
T223	UA1 receives Replay
T224	UA2 sends response to establish the connection
T225	UA1 sends ACK message
T18	Connection established and end-2-end path established, data will be transmitted by the data plane
T227	BYE message sent from UA1
T227	BYE message received by UA2
T19	Reservation finished and end-to-end path is no yet required, so release resources by UA1
T20	Reservation finished and end-to-end path is no yet required, so release resources by UA1

service can be decreased at the application to avoid congestion.

The Middleware: since the signaling path is established by the SIP protocol, the application starts sending the data to the middleware. Place P0 represents the input buffer of the middleware used to marshal data. The place P1 describes the main internal buffer of the middleware and it takes a number of tokens (N) representing the maximum number of messages that this queue can process at a given time. In many practical cases, the middleware retrieves the data from the upper layer and processes them into its proper queue then gives them to the underlying network layer. Therefore, transition T4, which has rate α_2, represents messages sent to the network via place P13 which represents the interface between the middleware and the network. Since we considered alleviating complexity of the service deployment, this hierarchical approach gives answer to challenge 2. We should note that at the receiver side the same abstraction of the middleware (main buffer, queue) are represented. Since this bloc acts just as a receiver the notification interface is not provided for simplicity seeks'. Therefore place P01 represent the incoming buffer of the receiver side middleware and places P11 and P21 show its main queue. Messages incoming from the network and traverse the middleware and, via transition T41, reaches the receiver application with distribution rate α_3. This pdf function models the delay taken from the middleware to process the data at the upper layer, and this rate depends on the type of the stream (multimedia, real-time,...) and the underlying operating system (real-time, not real-time).

The Control Plane: the management at this plane is mainly implemented by two components: the resource manager (called here Call Admission Control) and the resource allocator. Figure 2 shows the control plane in our model: the Call Admission control is located in bloc CAC, and the resource allocator is given in bloc RA. Table 3 and Table 4 illustrate the specifications and the semantics of both the places and the transition of the control plane: at the beginning of the simulation, available resources for the stream processing, presented by tokens in place P14 representing the length of the CAC queue. When a token is present in places P4, PC5 and/or PC6 with timeout mean duration $\frac{1}{\lambda_1}$ and $\frac{1}{\lambda_2}$ (we assume a pdf distribution), a call stream is coming to the CAC bloc and a stream will check for the availability of its resources requirements. In fact, the middleware component is responsible, via transition T4, for the specification of the QoS requirements of each flow incoming from the application.

Table 3. Specification of the places in CAC bloc

Transitions	Semantic
P4	Call admission
P5	Call from flow 1
P6	Call from flow 2
P14	CAC Queue
P15	Available resources
P16	Reserved resources

In a real case, An Service Level Specification (SLS) [4] contains exclusively the technical details specified by an SLA. It is essentially a translation of a SLA into the appropriate information necessary to configure network devices. Once a domain has agreed to honor the conditions set out in a SLS, it must be responsible for giving the guaranteed service to traffic specified in the SLS for the duration of the agreement. Hence, when a token arrives at place P4 by firing the timed transition T8 with timeout duration equal to $\frac{1}{\lambda_3}$, and then a call is incoming from a middleware with 'pdf' distribution and a Resource Allocation Request (RAR) is sent for resources in place P15 as outlined in the SLS. Indeed, the length of CAC queue in place P14 is updated and a Resource Allocator Answer (RAA) is returned to the host requesting to confirm whether the network has been reconfigured successfully according to the SLS in place P16 without exceeding the maximum length of queue indicated by the 'M' on the inhibitor arc between P14 and T8.

Table 4. Specification of the transitions in CAC bloc

Transitions	Semantics	Rates
T6	mean duration of timeout of flow 1	λ_1
T7	mean duration of timeout of flow 1	λ_2
T8	mean duration of timeout of CAC	λ_3
T9	Flow1 Call Forward	1
T10	Flow2 Call Forward	1
T11	Reservation Process of Flow 1	λ_4
T12	Reservation Process of Flow 2	λ_5

Once the call is admitted via transition T9 and T10 with an immediate priority, and there is enough available resources for a call, transition T11 and T12 are enabled. The reservation strategy need to be determined somewhere by the corresponding pdf rates of λ_4 and λ_5, respectively to decide whether it

is advantageous to admit an incoming call or not. Hence, the decision process for the resource allocation starts at places P8 and P9 after have been admitted by the CAC, and the place P16 is updated and the necessary resources are reserved. Finally, packets are sent to the buffer of the next DiffServ boundary nodes (places P10 and P11), and the traffic is transferred between the different nodes of the network with the respect of each traffic type. Transition T21 represents the bounder node which is directly connected to the host receiving packets. Once data are received by the middleware, the inverse process starts and the application is notified. For example, in Figure 2 the receiver is notified to receive multimedia stream including (audio and video) via the codec buffer. The information about the audio and video layers with stream identification and synchronization information are essential to the decoding and the subsequent rendering for each of them. So, before being presented at the upper layer, represented by the place *Receiver*, the synchronization have to be achieved.

following condition:

$$max(s_1, s_2, ..., s_n) \leq k \tag{3}$$

the marking is 's', that is, we note: Thus, the token count in places P_j is never exceeds k. Let T(s) a set of the enabled transitions in Figure 2, when

$$T(s) = \{t \in T, s_j \geq 1; for\ P_j \in I(t)\ and\ s_j = 0\ for\ P_j \in L(t)\} \tag{4}$$

In Figure 2, a transition t ∈ T-T(s) is disabled where the marking is 's', and:

$$\mathcal{G}(t) = \{s \in \mathcal{G} : t \in T(s)\} \tag{5}$$

We define S and S' the timed and the immediate markings, respectively:

$$S = s \in \mathcal{G} : T(s) \cap T' \neq 0 \tag{6}$$

$$S' = \mathcal{G} - S = s \in \mathcal{G} : T(s) \cap T' \neq 0 \tag{7}$$

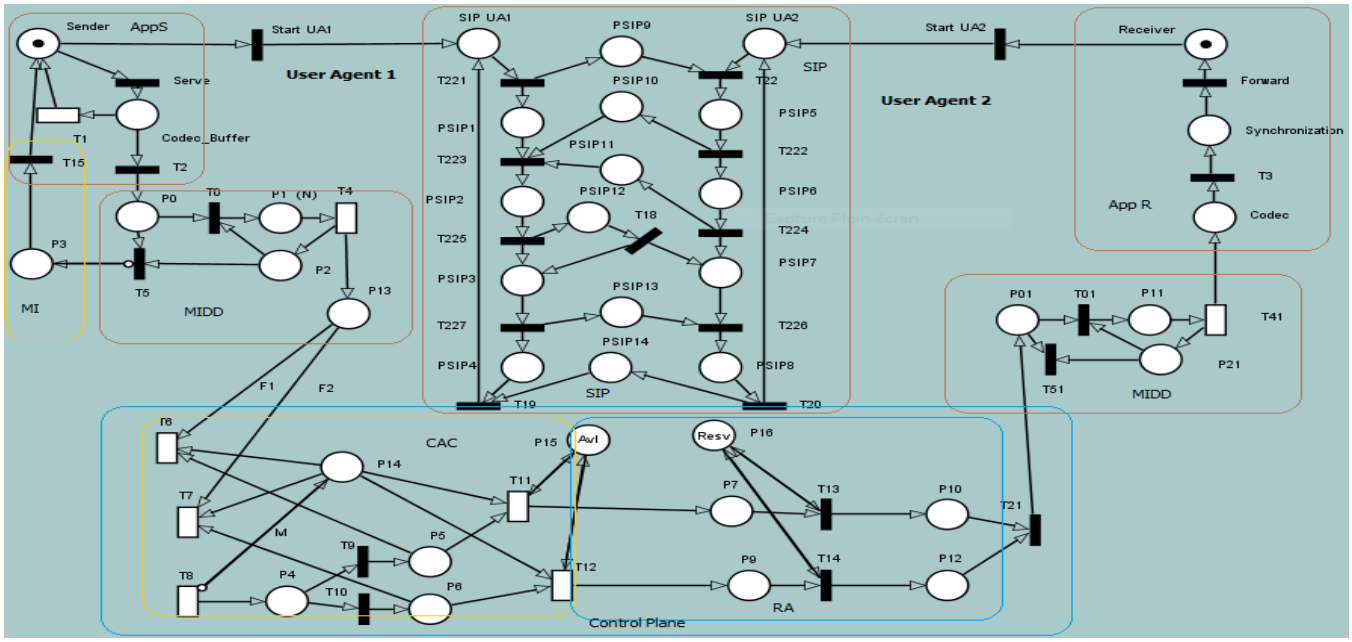

Figure 2. GSPN Model architecture

5.2. Analyzes and discussion

As we outlined before in this paper, concise specification of new marking probabilities can be assisted by numerical analysis. These probabilities determine the mechanisms by which a transition removes token from random set of its input places and deposit tokens in random set of its output places when it fires. In addition to definition given in section 2, we introduce the following: Let define by $\mathcal{G}$ the finite or infinite set of marking in our GSPN model. For

$$s \in \mathcal{G}; s = (s_1, s_2, ..., s_n) \tag{2}$$

where s_j is the number of tokens in place $P_j \in$ P. The GSPN model of Figure 2 is called K-bounded $k \geq 1$ and verify the

Since our GSPN model verify these conditions, any one of its markings changes when one or more enabled transitions fire. Let:

$$T^* \subseteq T(s), let\ p(s', s, T^*) \tag{8}$$

where p is the probability of new marking s' given the marking s and the transition set T^*, we can easily note the following:

$$s_j - \sum_{t \in T^*} 1_{I(t^*)}(P_j) \leq s' \leq s_j + \sum_{t \in T^*} 1_{I(t^*)}(P_j) \tag{9}$$

where j= 1..n, and 1 is the indicator function of the set A. **Global analysis:** the PN model is K-bounded and deadlock

free: this model works properly, and it takes into account the losses that can occur during communication, and the resulting behavior is live.

Local analysis: consider the middleware bloc in Figure 2, a token in place P0 corresponds to the job (message) in the system, the firing of the transition T0 corresponding to the event "arrival of message" and firing of timed transition T4 corresponds to the event "completion of the service" within the internal message queue of the middleware, and message is scheduled to be sent to the network. The middleware bloc (MIDD) can be modeled as in Figure 3:

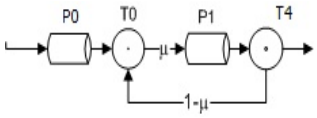

Figure 3. Middleware Queue model

T4: service completion in server 2 (Queue Manager)
T0: service completion in server 1 (Main Queue)

$$p(s,s,T4) = 1-\rho \quad (10)$$

$$p((s_1-1,s_2-1);s_1t_0 = \rho \quad (11)$$

$$for\ s=(s1,s2)\in\mathcal{G} \quad (12)$$

In this manner, the GSPN scheme modeling the message queue capturing the feedback mechanisms in the network queue, fulfilling the stability condition in equation 13 can be assumed:

$$\alpha = \frac{\rho}{1-\rho} \leq 1 \quad (13)$$

Furthermore, We have shown the SIP bloc is able to fulfill the end-to-end QoS signaling and providing the required QoS as needed by the protocol. When considering the control plane using the same formalism, it is easy to analyze the reservation strategy chooser by the CAC: preparing resources to be used by an active call take some time to be done, corresponding to the firing rates in transitions T6, T7 and T8 given in Table 4. As a result, the PN model is k-bounded, deadlock free, and live.

6. RELATED WORKS

This Section compare our activities on Middleware-based QoS provisioning with related works.

QoS management in reflective middleware: current research issues include adaptive and reflective middleware, and middleware for mobile and ubiquitous systems. For example, [6] presents a programmable networking approach to provide QoS as component based architecture. In our approach, we focus on how to use existing middleware and mapping them to provide the QoS for existing application. For instance, HLA and DDS-based existing applications used in large scale networks suffer from the lack of QoS provisioning in multi-domain networks like Internet.

Network QoS broker in middleware: Previous works focusing on integrating the signaling process into the QoS provisioning mechanisms. [3] focuses on message based signaling middleware for the control plane to offer per-class QoS. In real word, it does not suffice to deploy router to guarantee the QoS between two hosts at the control plane. likewise, [7] presented a network communication broker to provide per-class QoS for multimedia collaborative applications. This work, however, add an interface to the application and the middleware for the QoS notification when an event occur in the network and the application should be adapted to this modification. This approach has the advantage of working in plug and play like approach: interface can be used to adapt the application to the network context, and with a minimum of modification this model using simple signaling approach can provide also the 'On-demand' QoS.

Middleware-based Network QoS management: Earlier works [8][9] on integrating QoS with the middleware focused on how to add layer3 and layer2 services for CORBA based communication. This approach can be easily deployed in mono-domain network, where only one administrative domain can manage the whole network. If extending it to the internet, the Class of traffic specified at each host will be dropped by the network infrastructure (core and access routers) of other domains. Hence, the QoS provisioning should be enhanced by SIP signaling protocol to enforce resources reservation process during the signaling phase and before the data transmission. The goal of the model presented in this paper is twofold: at the service plane, application signaling using SIP allows the sender to contact the receiver to obtain its IP address and to agree the codec to be used; at the control plane, the network QoS provisioning aims to translate the application requests embedded within the SIP messages , via the middleware, to the network layer and coordinate the end-to-end path management and to enforce the end-to-end network resource reservation. Table 5 illustrates a comparison

Table 5. Comparison of the different approaches

Features	R1	R2	R3	Our approach
Application Adaptability	✓			✓
QoS specification	✓	✓	✓	✓
Resource allocation		✓	✓	✓
QoS guarantee			✓	✓
Multi-domain QoS				✓

of the related works presented in this Section vs. our approach. R1 refers to the QoS management in reflective middleware, R2 refers to the Network QoS broker in middleware, and R3 refers to the middleware-based network QoS management. Despite the adaptive and reflective approaches in R1 and R2 work well today when they receive the all the resources required, however, it has been shown that over-provisioning resources has several drawbacks in scalability and fail completely under the slightest anomaly [10-11]. Further, in R3 the QoS provisioning is considered in mono-domain architec-

ture. From the expectation of this Table, our approach aims to put all approaches together to provide adaptive framework which enables the end-to-end QoS provisioning over multi-domains independently for the underlying transport technologies. More, the stochastic delays in timed transitions allow the diverse DRE flows selecting their QoS requirements. Although the architecture generally works with standard SIP, it does not explicitly deal with QoS provisioning in core networks.

7. CONCLUSION

We have proposed using GSPN modeling a new model for QoS provisioning based middleware for multimedia and real time application. The following is the summary of the lessons learned: (1) A PN model based approach provides a high level view and the key concepts actions and protocol to be used to coordinate the different technologies for better QoS provisioning in multi-domain infrastructure, and probabilistically time delay is introduced transitions to allow performance evaluation and resolving scheduling problems in DRE. (2) the functional description of this model is based on three concepts: Provisioning, Invocation and operation. (3) For middleware based communication, both per-class and per-flow are needed to support the QoS on demand: after signaling and control phases, the data should be transferred using the appropriate transport protocol. However, this approach still need further research. for example, when including virtual data path with defined QoS parameters, which is the future direction of this work.

Acknowledgment

This research is supported by the French FUI-DGE (Single Inter-Ministerial Fund of the Directorate General for Enterprise) program within the network simulation Platform (PLATSIM).

References

[1]Rosenberg, J. et al., '*SIP: Session Initiation Protocol*', RFC 3261, 2002.

[2]Diaz. M, *'Petri Nets Fundamental Models, Verification and Applications'*. Published JAN 2010

[3]G. Teodoraet al., '*A Session Initiation Protocol based Middleware for Multi-Application Management*'. IEEE International Conference on Communications - ICC, Multimedia Communications & Home Services Symposium (Glassgow, UK, 2427 06 2007).

[4]P. Nanda and A. Simmonds,'*Providing end-to-end guaranteed quality of service over the Internet: a survey on bandwidth broker architecture for differentiated services network*', 4th CIT 2001, Berhampur, India, 2023 Dec 2001, pp.211216.

[5]http://www.voiptroubleshooter.com/problems/plc.html

[6]L. Capra et al.,'*Reflective Middleware Solutions for Context-Aware Applications*', 3th International Conference on Metalevel Architectures and Separation of Crosscutting Concerns, pp 126-133, 2001

[7]Chi Z, Sadjadi M, Weixiang S, Raju R, and Yi, D,'*A user-centric network communication broker for multimedia collaborative computing*', CollaborateCom 2006.7-20 Nov. 2006, pp1-5

[8]Dasarathy et al., *Network QoS Assurance in a Multi-Layer Adaptive Resource Management Scheme for Mission-Critical Applications using the CORBA Middleware Framework*,11th IEEE-RTAS, p.246-255, March 07-10, 2005

[9]Balakrishnan D et al., *Adaptive network QoS in layer-3/layer-2 networks as a middleware service for mission-critical applications*,The Journal of Systems and Software, 2007

[10]D C. Schmidt et al, Middleware R&D Challenges for Distributed Real-time and Embedded Systems, ACM SIGBED, Volume 1 Issue 1, April 2004

[11]D C. Schmidt, *R&D Advances in Middleware for Distributed, Real-time and Embedded Systems*, Communications of the ACM, Volume 45, Number 6, June 2002, edited by Gul Agha.

[12]Shih-Hsi Liu et al, *QoS-UniFrame: a Petri net-based modeling approach to assure QoS requirements of distributed real-time and embedded systems*, 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems, pp 202-209, 2005. ECBS '05

[13]Ana Cavalli et al, *New approaches for passive testing using an Extended Finite State Machine specification, Information and Software Technology*, V 45, Issue 12, 15 September 2003, Pages 837-852

[14]Sandeep N et al, *Generators for Synthesis of QoS Adaptation in Distributed Real-Time Embedded Systems*, ACM SIGPLAN/SIGSOFT conference on Generative Programming and Component Engineering, 2002

[15]Sandeep N et al, *Constraint-based design-space exploration and model synthesis*, International conference on embedded software, EMSOFT 03, V 2855, pp 290-305, 2003

[16]John S. Kinnebrew et al, *Intelligent Resource Management and Dynamic Adaptation in a Distributed Real-time and Embedded Sensor Web System*, Proceedings of the 12th International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC '09), Tokyo, Japan, March 17-20, 2009.

[17]Nilabja Roy et al, *Toward Effective Multi-capacity Resource Allocation in Distributed Real-time and Embedded Systems*, Proceedings of the 11th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing, Orlando, Florida, May 5-7, 2008.

[18]Prakash Manghwani et al, *End-to-End Quality of Service Management for Distributed Real-Time Embedded Applications*, Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05), Volume 03

[19]Camarillo G. et al, *Integration of Resource Management and Session Initiation Protocol (SIP)*, RFC 3312, http://www.rfc-editor.org/rfc/rfc3312.txt

[20]http://www.tu-ilmenau.de/fakia/8086.html