International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 8, August 2017)

Network Protocol Modeling using Colored Petri nets

Veena Bharti[1], Sachin Kumar[2]
[1]Raj Kumar Goel Institute of Technology, Ghaziabad, UP, India
[2]Ajay Kumar Garg Engineering College, Ghaziabad, UP, India

***Abstract*— Network protocol is modeled using CPNs with the aid of Design/CPN. The model is based on the description of the protocol. This paper includes a detailed description of the CPN model, the scope and assumptions of the model and the modeling decisions. The CPN model of RSVP has benefited from several discussions and detailed comments from Professor Billington. In this paper, the main features of RSVP are considered.**

***Keywords*— CPN, OPWA, Service abstraction**

I. INTRODUCTION

As the provision of the Internet services has become more sophisticated, the complexity of the Internet protocols has increased. Changes to the Internet can be a major undertaking and have a big impact for all the millions of users connected to the network around the world. Thus, new protocols to be added to the architecture must supplement it but not replace it. In order to achieve this goal, the new protocols may have to be more complex. Therefore, protocol engineering techniques may be required to design high-quality protocols. The service specification describes the service that is provided to the user. The aim of this paper is to propose a service specification by performing *service abstraction.* The abstraction consists in taking the protocol specification and identifying the service required RSVP[1] will be modeled using CPNs. CPNs allow the creation of models at different levels of abstraction. Thus, the aim of this paper is to model RSVP at a level of abstraction that captures the functional properties which need to be proved and allows analysis tools to be used, given the limitations on computer resources and the complexity of the protocol. RSVP can operate on either unicast or multicast networks. However, only a unicast network is considered. Also, the network includes a sender and a receiver host and a router, which connects them as shown in Figure 1.

Figure 1: RSVP network topology.

RSVP may communicate with several components of the IntServ architecture RFC 2205 [2] specifies the interfaces between RSVP and each of those components. The RSVP model only considers the interaction between RSVP and the application. Multiple RSVP sessions can be open simultaneously. However, without loss of generality, just one session is modeled to study the functional behavior of RSVP, since RSVP treats each session independently [6].Several data flows can be generated by different applications running at the sender host. Having multiple real-time and multimedia applications running concurrently on a single machine may be uncommon given the performance issues related to these applications. In order to simplify the analysis of RSVP, only one data flow is considered in the model. The above restrictions impose some limitations on the number of RSVP features that can be modeled and simplify the operation of RSVP.

The restrictions also impose limits on the operation of reservation confirmations. Any router in the network can generate a reservation confirmation. However for a unicast network, only end-to-end confirmation (i.e. the confirmation is generated by the sender host) occurs.

II. ASSUMPTIONS

The CPN model of RSVP is based on the following assumptions, which are related to the network or to RSVP.

A. Network Considerations

The considerations about the underlying network, which connects the sender host with the router and the router with the receiver host, are described as follows.

Perfect link

The link or network, which connects two adjacent nodes, is assumed perfect, so RSVP messages cannot be corrupted or have bit errors. Therefore, the checksum procedure is not required.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 8, August 2017)

Packet losses

Since RSVP is running on top of IP, which is not a reliable protocol, messages may be lost, duplicated and overtaken. *Path* and *Resv* [4] refresh messages deal with occasional loss of RSVP messages. Although message losses have not been considered in the model, the mechanisms for dealing with losses are modeled. RSVP messages can be duplicated. RSVP does not have any mechanism to deal with message duplication. For example, duplicated *Path* and *Resv* messages can be treated as refresh messages.

Overtaking

Messages sent from one node (e.g. an end host or a router) to another (i.e. the next node calculated by the routing protocol) may not arrive at their destinations in the same order they were delivered. Some of the reasons for overtaking on adjacent nodes are that the underlying network does not guarantee message order, and the scheduling mechanism may assign different RSVP messages to different queues. These events are unlikely to happen. Hence message overtaking is only considered.

B. Protocol Assumptions

The following assumptions are related to the description of RSVP given in RFC 2205 .

Reservation Initialisation

The receiver can start sending *Resv* messages at any time, it does not need to wait for the first *Path* message to arrive. If a *Resv* message arrives at a router where there is no path state information, a *ResvErr* message is sent back to the receiver. In order to avoid this problem, it is assumed that the receiver waits for the first *Path* message before starting to send any *Resv* message.

RSVP Requests

If is not clear from RFC 2205 [8] whether RSVP can distinguish between different sender or reservation requests, which travel in the messages and are generated as a result of a RSVPSender. Req or RSVP-Reserve.Req service primitive occurrence, for example, by using a request identification number. This can be useful when the user generates two or more requests with the same values of the parameters, such as identical QoS[5] information (e.g. data rate) for the same data flow and the network needs to process them as different requests. This paper assumes that these requests can be differentiated somehow, for example, by using one field in the corresponding RSVP object .

Release Indications

RFC 2205 [7] establishes that tear messages (*PathTear* or *ResvTear*) are generated because either the cleanup timer expires or the user leaves the session . However, it is not clear in RFC 2205 [8], if RSVP can distinguish between a *PathTear* or *ResvTear* message generated as a result of the expiration of the cleanup timer (network release) and one generated as a result of a user leaving the session (user release). This is necessary to generate the correct user release sequences. Otherwise, a release indication (i.e. RSVPSenderRel.Ind or RSVPReceiverRel.Ind) may occur without any occurrence of the corresponding release request (i.e. RSVPSenderRel.Req or RSVPReceiverRel.Req) . Also, multiple release indications (i.e. RSVPSenderRel.Ind or RSVPReceiverRel.Ind) may occur as a result of multiple expirations of the cleanup timer (this can happen, for example, if the network is congested).

Figure 2 shows an example of the problem. We assume that the path cleanup timer expires at the router. The router will generate and send a *PathTear* message to the receiver and a RSVPSenderRel.Ind service primitive will occur, but no RSVP-SenderRel.Req service primitive has occurred yet. Then, the router receives a path refresh message from the sender and re-establishes the path state information. After that, the path timer expires again and the sequences of events previously described, including another occurrence of the RSVPSenderRel.Ind service primitive, are repeated. This paper assumes that RSVP can distinguish between tear messages generated as a result of user release and network release.

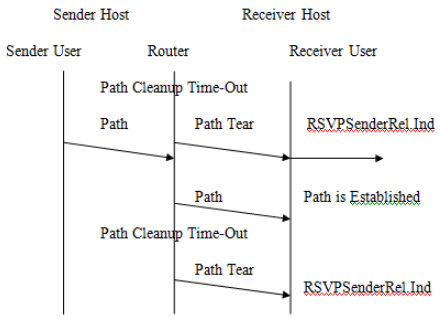

Figure 2: Diagram illustrating the occurrence of release indications without requests and multiple release indications.

III. Modeling Decisions

Several key decisions about how to model the features of RSVP have been taken. They are summarized as follows.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 8, August 2017)

A. RSVP Message Parameters

The structure of RSVP messages is complex. Each RSVP message may include more than one object, which may contain more than one field. Most of these parameters are not required to study the functionality of RSVP. The parameters considered in the model are the traffic specification, TSpec (or traffic characteristics) and the flow specification, FSpec (or flow characteristics). The traffic and flow specifications are important to model dynamic changes of the traffic and flow characteristics. The traffic and flow specifications include several fields . In the model, these fields do not need to be considered independently. Instead they are represented by abstract values. For example, the traffic specification may have the value Ta and the flow specification the value Fa . The session, RSVP_Hop, style, filter specification, and sender template objects are not necessary given the scope of the paper(See Section 2), which considers a simple topology (one sender and one receiver), one session and one data flow. The time values object is not considered because time is not included in the model, instead time-related functions, such as cleanup and refreshes, are modeled in a non-deterministic way. The AdSpec, policy and integrity objects are not required because the OPWA, policy control and integrity functions of RSVP are not modeled. The existence of the error specification object is irrelevant given the level of abstraction. Finally, the *Resv Confirm* object is not required because the reservation confirmation feature is compulsory in the model.

B. State Information

RFC 2205 does not include any state diagrams or tables, which indicate the possible states in which a RSVP entity (e.g. sender) can be in. The states of the RSVP entities are represented by places , which include some status information and RSVP parameters. The status of each entity has been derived based on the interpretation of RSVP functions described in RFC 2205 [6]. The path and reservation state information is generated based on some of the parameters carried in the RSVP messages including the traffic and flow specifications. Since the flow and traffic specification objects are the only parameters required in this thesis, they are the only parameters that need to be part of the states of the RSVP entities. In a real implementation, some of the fields in the traffic and flow specifications may not be included as part of the state information.

However, in order to simplify the model, the parameters carried in the messages are included (when required) in the state information without changing or disassembling them. For example, if a *Resv* message carrying a flow specification parameter (*fspec*) arrives at the router, where there is no reservation state, the state of the router is updated by assigning the *fspec* to a variable, which represents the state

C. Sender and Reservation Requests

The sender and reservation requests generated by the sender and receiver users, respectively, may not have the same structure as the corresponding requests carried in the RSVP messages. However, the procedures intended to convert the user requests into the corresponding parameters carried in RSVP messages are ignored.

D. Relationship with the Integrated Services Components

RSVP interacts with the traffic control components (i.e. packet scheduler, admission control and classifier) of the IntServ architecture in order to set up, update, and remove reservation and data traffic information [9]. The interaction with these components is ignored since it is not relevant for verification of RSVP functionality.

E. Model Hierarchy

The detailed model of RSVP is illustrated in Figure 3. The hierarchical view has been designed based on the network topology represented in the *RSVPNetwork* page and the functionality of RSVP entities at each node represented by the *Sender*, *Router* and *Receiver* pages and their corresponding sub-pages.

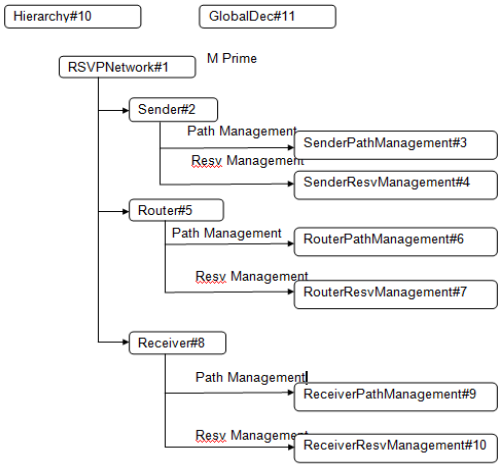

Figure 3: Hierarchy page.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 8, August 2017)

The pages located at the lower level of the hierarchy correspond to the major functions of RSVP. The *Path* and *Resv* management pages include transitions that model the establishment, refreshment, release and error control of paths and reservations. Similarly to the RSVP Service Specification model, each page at the lower level of the hierarchy may include one or more of the following types of transitions: service primitive, discard and protocol transitions. A discard transition handles situations where a RSVP message is received but the state of the entity indicates that the message cannot be processed. A protocol transition represents a RSVP function (e.g. path refresh). The GlobalDec page includes the colour sets, variables, and functions .

F. Global Declaration

Figure 4 to Figure 8 show the colour sets, variables, and functions from the global declaration node. It is divided into the following sections: *states of RSVP entities RSVP messages, control/flags, variables* and *functions*.

States of RSVP Entities

This section defines the states of the three RSVP entities considered in the model (i.e. the sender, receiver and router) and is shown in Figure 4. The colour set ParameterValues is an enumeration type, which represents abstract values for both the traffic specification (*tspec*) and flow specification (*fspec*) parameters. The possible values are Ta, Tb, Fa, Fb and E (empty). The colour set STSpec is a subset of ParameterValues and represents the traffic specification stored as part of the path state information. The colour set SFSpec is a subset of ParameterValues and represents the flow specification stored as part of the reservation state information. The colour set Status is an enumeration type, which defines the states of the RSVP entities. The values of this colour set have the following meanings:

1. SESSION: the sender or receiver has opened a session, but any data flow information or reservation has not been established yet.

2. IDLE: there exists neither data flow information nor reservation installed in the router.

3. WAITINGRESV: a request with the sender's data flow information has been accepted by the entity and sent (if the entity is not the receiver) but no reservation request has been received yet.

4. RESVREADY: a reservation request has been accepted and sent (if the entity is not the sender).

5. RESVCONFIRMED: a reservation has been established and a confirmation has been received.

6. CLOSED: the sender or the receiver has left the session.

Three subsets of the colour set Status have been defined: the SenderStatus, RouterStatus and ReceiverStatus. They indicate the possible status of the sender, router and receiver, respectively. The colour sets SenderState, RouterState and ReceiverState represent the states of the sender, router and receiver entities respectively. Each of them is the product of the status of the corresponding entity (i.e. SenderStatus, RouterStatus and ReceiverStatus respectively), the STSpec and SFSpec colour sets.

```
(* ============= States of RSVP entities =========== *)
color ParameterValues = with E|Ta|Tb|Fa|Fb;
color STSpec = subset ParameterValues with [E,Ta,Tb];
color SFSpec = subset ParameterValues with [E,Fa,Fb];
color Status = with SESSION| IDLE| WAITINGRESV|
RESVREADY| RESVCONFIRMED|
CLOSED;
color SenderStatus = subset Status with
[SESSION,WAITINGRESV,RESVREADY,CLOSED];
color ReceiverStatus = subset Status with
[SESSION,WAITINGRESV,RESVREADY,
RESVCONFIRMED,CLOSED];
color RouterStatus = subset Status with
[IDLE,WAITINGRESV,RESVREADY];
color SenderState = product SenderStatus * STSpec *
SFSpec;
color ReceiverState = product ReceiverStatus * STSpec
* SFSpec;
color RouterState = product RouterStatus * STSpec *
SFSpec;
```

Figure 4: States of RSVP entities.

RSVP Messages

The colour sets representing the RSVP messages are shown in Figure 5. The colour sets TSpec and FSpec represent the parameters that may be carried in the RSVP messages and are the traffic specification and flow specification, respectively. They are subsets of the colour set ParameterValues.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 8, August 2017)

The colour set TearMsgType is intended to distinguish between a *PathTear* or *ResvTear message* generated as a result of the sender or receiver leaving the session (USER_REL) and a path or reservation cleanup time-out (NETWORK_REL) .

The seven RSVP messages are considered in the model.

```
(* ================= RSVP Messages ================= *)
color TSpec = subset ParameterValues with [Ta,Tb];
color FSpec = subset ParameterValues with [Fa,Fb];
color TearMsgType = with NETWORK_REL|USER_REL;
color ResvTear = product TearMsgType * FSpec;
color PathTear = product TearMsgType * TSpec;
color UpstreamMessages = union patherror: TSpec + resvtear: ResvTear + resv: FSpec;
color DownstreamMessages = union path: TSpec + resverror: FSpec + resvconf: FSpec +
pathtear: PathTear;
```

Figure 5: RSVP message definition.

The colour sets UpstreamMessages and DowstreamMessages represent the messages that travel from the receiver to the sender and from sender to receiver, respectively. The colour set UpstreamMessages is the union of the colour sets representing the messages that travel upstream (i.e. PathError, ResvTear and Resv) and are explained as follows:

1. patherror: models a *PathErr message*. It is represented by the TSpec colour set, which models the TSpec parameter carried in the *Path message* that caused the error.

2. resvtear: models a *ResvTear message*. It is represented by ResvTear colour set. It is the product of the TearMsgType and FSpec colour sets. The FSpec colour set models the value of the current FSpec, which is being torn down.

3. resv: models a *Resv message* and is defined as the colour set FSpec. The FSpec colour set represents the requested resources. The colour set DowstreamMessages is the union of the colour sets representing messages that travel downstream (i.e. Path, ResvError, ResvConf and PathTear) and are explained as follows:

1. path: represents a *Path message* and is defined as the colour set TSpec. The TSpec colour set represents the requested traffic specification of the data flow.

2. resverror: represents a *ResvErr message* and is defined as the FSpec colour set. The FSpec colour set represents the flow specification carried in the *Resv message*, which caused the

error.

3. resvconf: represents a *ResvConf message* and is defined as the FSpec colour set. The FSpec colour set represents the requested flow specification, which is being confirmed.

4. pathtear: represents a *PathTear message* and is the product of the TearMsgType and TSpec colour sets. The TSpec colour set represents the traffic specification, which is being torn down.

Control and Flags

The control/flags section defines the set of colour sets used to control RSVP operation and the service primitive occurrence at the protocol level and is shown in Figure 6.

```
(* ================ Control/Flags ================== *)
(* Remote User flag *)
color UserInd = with USR|NOUSR;
(* Indication primitive flag *)
color Flag = int with 0..1;
color FSpecXFlag = product FSpec * Flag;
color TSpecXFlag = product TSpec * Flag;
```

Figure 6: Control and flags.

The colour set UserInd is an enumeration type and indicates whether the user at the other end has left the session or not. It is used to avoid situations where some events may occur after an indication that the user at the other end has left the session has been received. For example, only the RSVP-ReceiverRel.Req service primitive can occur after a RSVPSenderRel.Ind service primitive has occurred . The value USR is the initial state and indicates that the user at the other end is still part of the session, while the value NOUSR indicates that it has left the session. The colour set Flag is an enumeration type and has two values *0* and *1*. The value *0* indicates that the flag is OFF and the value *1* indicates that the flag is ON. The colour set FSpecXFlag is a product of the colour sets FSpec and Flag and indicates whether a RSVP-Reserve.Ind service primitive, which includes the requested flow specification, FSpec, has occurred (value of the flag is ON) or not.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 8, August 2017)

It is used to avoid multiple occurrences of this primitive with the same flow specification. The colour set TSpecXFlag is the product of the colour sets TSpec and Flag and indicates whether a RSVPSender.Ind service primitive, which includes the requested traffic specification, TSpec, has occurred (value of the flag is ON) or not. It is used to avoid multiple occurrences of this primitive with the same traffic specification.

Variables

Variables are shown in Figure 7. The variable *sta* is of the colour set Status and contains the status of either the sender or receiver. The variables *tspec* and *tspec1* are of the colour set STSpec and contains the values of the traffic specification, which is either part of the states of the sender or receiver or carried in the *Path*, *PathError* or *PathTear* messages. The variables *fspec* and *fspec1* are of the colour set SFSpec and contain the values of the flow specification, which is either part of the state of the sender or receiver or carried in the *Resv*, *ResvErr*, *ResvConf* or *ResvTear* messages. The variable *flag* is of the colour set *Flag* and contains the value of the flag (i.e. ON/OFF). The variable *rmtusrind* is of the colour set UserInd and indicates whether the remote user has left or not. The variable *ttype* is of the colour set TearMsgType and contains the type of either the *PathTear* or *ResvTear* messages. The variable *msg* is of the colour set DownstreanMessages and contains a message that travels from the sender to the receiver.

```
(*                ================                Variables
================ *)
var sta: Status;
var tspec,tspec1: STSpec;
var fspec,fspec1: SFSpec;
var flag: Flag;
var rmtusrind: UserInd;
var ttype: TearMsgType;
var msg: DownstreamMessages;
```

Figure 7: Variables.

Functions

The functions are used to simplify guard inscriptions and are shown in Figure.8. The pathexists function returns true if the status of the sender indicates that there is some path state information available in the node. A resvexists function returns true if the status of the receiver indicates that there is some reservation state information available in the node.

```
(*                ================                Functions
================== *)
fun pathexists (sn) = (sn = WAITINGRESV) or else (sn
= RESVREADY) or else (sn = RESVCONFIRMED);
fun resvexists (sn) = (sn = RESVREADY) or else (sn =
RESVCONFIRMED);
```

Figure 8: Definition of the functions.

IV. RSVP Network Page

The *RSVPNetwork* page (see Figure 9) is the top-level page of the model. It shows the interaction between the RSVP nodes. The three transitions shown in the figure (i.e. Sender, Router, and Receiver) represent the RSVP entities at each node and are shown as substitution transitions. The place SenderUser has the colour set TSpec and represents the traffic characteristics of the data flow requested by the sender user. The place ReceiverUser has the colour set FSpec and represents the flow specification requested by the receiver user. The places SenderUser and ReceiverUser are included in the model because of the protocol assumptions described in Section. The places SOutgoingMsgs and RIncomingMsgs have the colour set DownstreamMessages. They have markings consisting of tokens that represent RSVP messages travelling downstream (i.e. from the sender to the receiver), such as a *Path* message. The places ROutgoingMsgs and SIncomingMsgs have the colour set UpstreamMessages. They have markings consisting of tokens that represent RSVP messages travelling upstream (i.e. from the receiver to the sender), such as a *Resv* message.

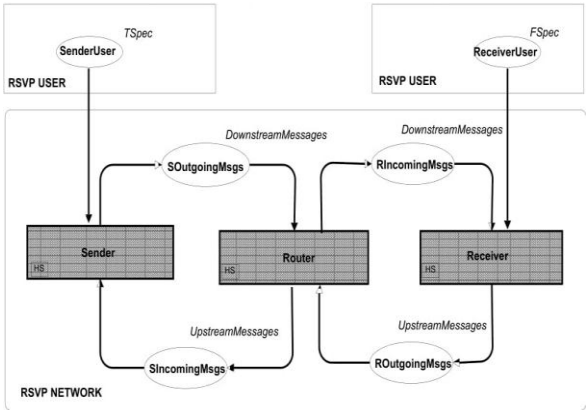

Figure 9: Top-level page showing the RSVP network and end users.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 8, August 2017)

V. RSVP SENDER

The RSVP-Sender page (see Figure 10) is located at the second level of abstraction and includes the major functions that are performed by the RSVP sender entity. All the transitions are hierarchy transitions (as denoted by the HS-tags) and their names are closely related to the RSVP functions. A brief description of the substitution transitions are given as follows:

1. *PathManagement:* models the establishment, refreshment, error control and release of the

path state information.

2. *ResvManagement:* models the establishment, refreshment and release of the reservation state information.

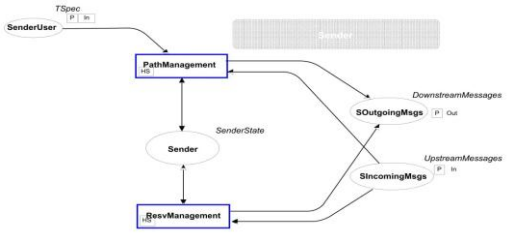

Figure 10 : Sender page.

The place Sender has the colour set SenderState and models the status of the sender together with the path and reservation state information. The other places are port places of the socket places 7.

VI. RSVP ROUTER

The Router page (see Figure 11) is located at the second level of abstraction and includes the major functions, which are performed by the RSVP router entity. The substitution transitions (denoted by the HS-tags) are:

1. PathManagement: models the establishment, refreshment, error control and release of the path state information.

2. ResvManagement: models the establishment, refreshment, error control and release of the reservation state information.

The place Router has the colour set RouterState and models the status of the router together with the path and reservation state information. The other places are port places of the socket places 7.

In this sections, the sub-pages associated with the substitution transitions are described.

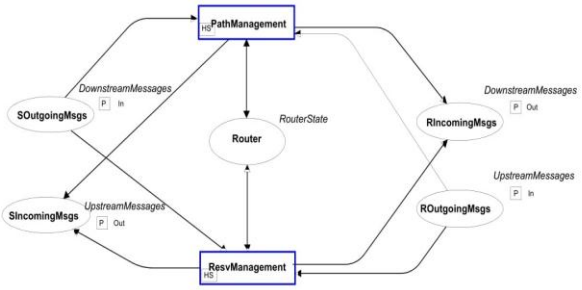

Figure 11: Router page.

VII. RSVP RECEIVER

The Receiver page (see Figure 12) includes the major functions performed by the RSVP entity at the receiver node. The substitution transitions are:

1. PathManagement: models the establishment, refreshment and release of the path state information.

2. ResvManagement: models the establishment, refreshment, error control and release of the reservation state information.

The place Receiver has the colour set ReceiverState and models the status of the receiver together with the path and reservation state information. The other places are port places of the socket places.

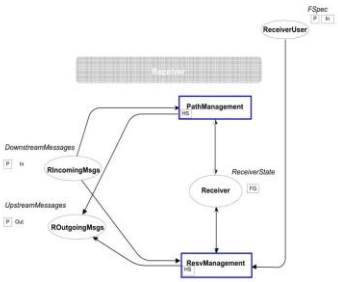

Figure 12: Receiver page.

VIII. SUMMARY

Coloured Petri Nets have been used to model the main features of RSVP based on a number of simplifying assumptions. The simplest representative network topology (one sender, communicating with a receiver via a single router) and unicast operation is assumed. This is to make the model tractable and to verify later that RSVP will operate correctly under the most ideal of conditions.

REFERENCES

[1] https://en.wikipedia.org/wiki/Resource_Reservation_Protocol.

[2] Journal of Electrical and Computer Engineering Volume 2017 (2017).

[3] Slavomír Šimoňák "Verification of Communication Protocols Based on Formal Methods Integration" Acta Polytechnica Hungarica, Vol. 9, No. 4, 2012 .

[4] [Manju Sanghi "Improved G-3PAKE Protocol with Formal Verification", International Journal of Information & Network security Vol.1, No.1, April 2012, pp. 1 - 8 ISSN: 2089-3299.

[5] 3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; USIM (Release 4) 2009.

[6] 3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; USIM (Release 10) 2011.

[7] Mohamed Sabt, Mohammed Achemlal, Abdelmadjid Bouabdallah," Over-the-Internet: Efficient Remote Content Management for Secure Elements in Mobile Devices", HAL Id: hal-01118705 https://hal.archives- ouvertes.fr/hal-01118705 Submitted on 19 Feb 2015.

[8] Murata T., Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE, Vol. 77, No. 4, April, 2010 pp 541-580.

[9] www.omgsysml.org/INCOSE_IS_2016_paper_Application-of-a-Layered-Interface-by PM Shames -2016.