RESEARCH INVENTY: International Journal of Engineering and Science
ISSN: 2278-4721, Vol. 1, Issue 9 (November 2012), PP 33-38
www.researchinventy.com

Modelling of RSVP Protocol using Coloured Petri Net

[1]Veena Bharti, [2]Sachin Kumar
[1]*(Assistant Professor,MCA Dept.,Raj Kumar Goel Institute Of Technology, Ghaziabad ,India)*
[2]*(Associate Professor,CS Dept.,Ajay Kumar GargEngg.College , Ghaziabad ,India)*

Abstract: *The main formal technique used for RSVP specification and verification is CPNs. In this paper, their functionality is described briefly by using RSVP without giving any formal definitions. The paper gives an informal introduction to the components, dynamic behaviour, occurrence and enabling mechanism of RSVP using CPNs.*

Keywords: *CPN, RSVP, service specification.*

1. INTRODUCTION

Formal methods encompass a variety of modelling techniques based on mathematics, which are applicable to computer systems. They are useful in the construction and maintenance of complex communication protocols and allow protocol specifications to be formally analysed and verified. A wide range of formal methods have been developed, however, this paper is not intended to present a comprehensive treatment of them. Instead, it is focussed on *Coloured Petri Nets* (*CPNs*) [1].There are several reasons for using CPNs for modelling and analysis of RSVP. The main ones are summarised as follows. Petri Nets are a mature technique. That is shown in the thousand of journal papers and research reports generated in more than 30 years of theoretical and practical work. Petri Nets are a well-defined graphical tool, which allows formal analysis. This paper provides an informal introduction to CPNs and the computer tool, which supports its practical use, Design/CPN [3].

2. PETRI NETS

Petri Nets [6] are a graphical technique that is based on a solid mathematical foundation and can be used for describing and studying several information processing systems characterised by being concurrent, asynchronous, non deterministic, parallel and distributed. This paper is intended to give a brief overview of the main concepts and definitions related to Petri Nets by the means of some examples. Ordinary Petri Nets are also called Place/Transition systems (PT-systems). The main components of PT-nets are the *places*, *transitions* and *arcs* . Places are drawn as ellipses or circles while transitions are drawn as rectangles. Arcs are directed either from a place to a transition or from a transition to a place and may have a *weight* (positive integer) associated with them (the default weight of an arc is one and is not shown). Depending on the system to be modelled, places and transitions can have different interpretations [7]. For example, a place can represent a condition and a transition an event. Also, a place can represent resources and a transition a task or job, which requires those resources. Finally, a place can represent the state of the system and a transition, an action that can be taken, based on that state.

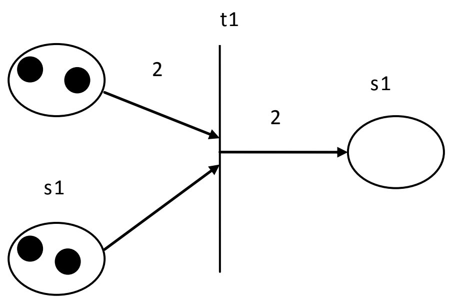

Figure 1: An example of Petri Nets.

A place can contain one or more *tokens*. They are drawn as black dots (see Fig. 1). A distribution of tokens on the places is called a *marking* (i.e. marking of the net). The initial state of the system is called the initial marking. A marking of a place indicates the number of tokens on a particular place. The dynamic behaviour of a PT-system can be seen as the state or marking of the net changing according to transition occurrences. A transition may have *input places* connected by incoming arcs and *output places* connected by outgoing arcs. A transition is *enabled* (i.e. it can occur) if the marking of each input place consists of as many tokens as indicated by the weight of the input arc, which connects the place with the transition. The occurrence of an enabled transition removes tokens from the input places and adds tokens to the output placesThe number of removed tokens from each input place corresponds to the number of tokens indicated by the weight of the (input) arc, which connects the place and the transition. The number of added tokens to each output place corresponds to the number of tokens indicated by the weight of the (output) arc, which connects the transition with the place. Fig. 2 presents an example of the occurrence of the transition t1 in Figure 1. After transition t1 occurs, two tokens are removed from place S1 and one from place S2, and two tokens are added to place S3.

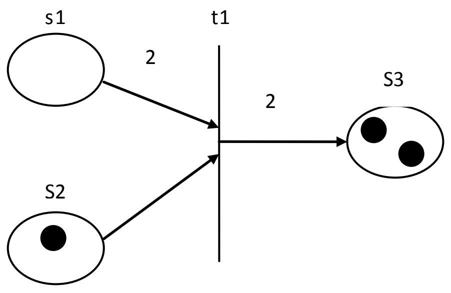

Figure 2: An illustration of the occurrence of transition t1**.**

3. Coloured Petri Nets

As systems become more complex, Petri Net models can become very large, complex, and probably unreadable [5]. This problem has been overcome by introducing new kinds of Petri Nets, called *High-Level Petri Nets* [5]. *Coloured Petri Nets (CPNs)* are high-level nets. They incorporate some definitions, such as data types and data values processing found in programming languages [5][9]. In this section, CPNs are introduced through a small example of the service specification for a communication protocol. A more detailed explanation and the formal definitions of CPNs can be found in [5]. The example is a simplified version of the RSVP service specification model .Some modifications have been introduced to the original model to incorporate all the concepts of the CPN language, which are used in this paper.

3.1 Example: Simplified Version of the RSVP Service Specification

Fig.3 shows an overview of the system. It consists of two users, the sender and receiver that are connected through a network that supports RSVP.

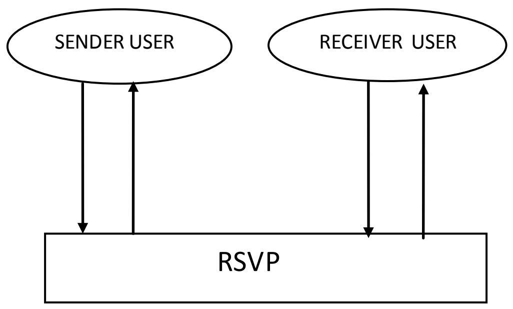

Figure 3: Overview of the RSVP service specification.

The sender and receiver communicate to request some QoS guarantees, in the form of resource reservations. The underlying protocol is receiver-oriented. The data packets will travel from the sender to the receiver, but they are not shown in the model. Thus, the sender that wishes to get some QoS guarantees for the data flow, signals its intent by sending a request including the traffic characteristics of the data flow to the receiver.Once the receiver gets the message it can send a reservation request including the details of the reservation to the sender. The network may reject the reservation request because there are not enough network resources to satisfy it. In that case, an error message is sent back to the receiver. Otherwise, the request arrives at the sender, and the sender user is notified about the current reservation.

3.2 CPN Model

As Petri Net models, CPN models are created as graphical drawings. Fig.4 shows the CPN model of the example. It is divided into five parts: the sender user, the sender application/RSVP interface, the RSVP service provider, the receiver application/RSVP interface and the receiver user. The basic components of the CPN model are described as follows.

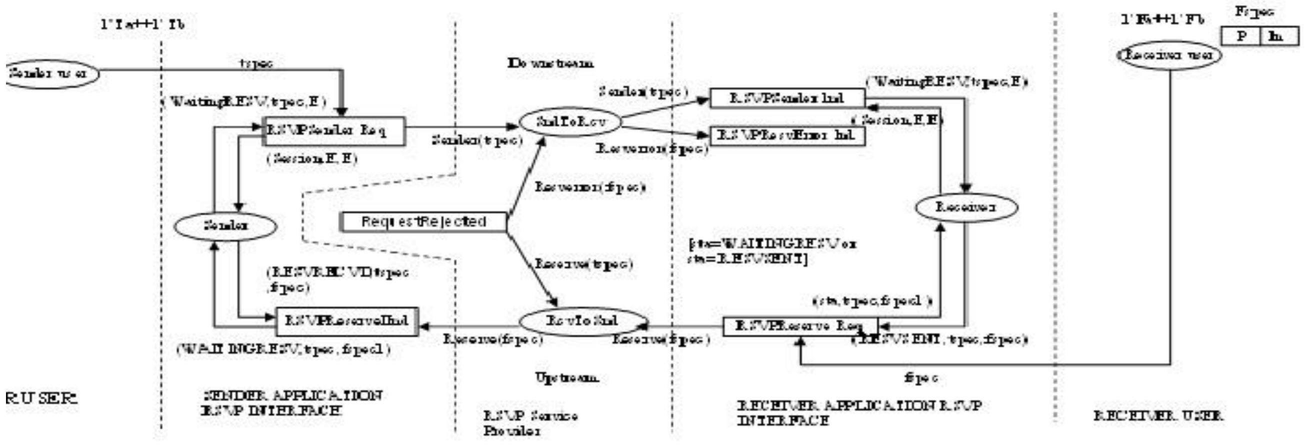

Figure 4: Simplified Model of the RSVP Service Specification

3.2.1Places

There are six places drawn as ellipses. The Sender and Receiver places represent the state of the sender user/RSVP interface and receiver user/RSVP interface, respectively. The places SndToRcv and RcvToSnd represent the RSVP service provider, one for each communication path (i.e. from the sender to the receiver - SndToRcv and from the receiver to the sender - RcvToSnd). A communication path includes all the intermediate devices along the route from the sender to receiver and vice versa. Finally, the places SenderUser and ReceiverUser represent the users of the RSVP services.

3.2.2Types

Each place has an associated *type* or *colour set* which determines the type of data the place may contain. It is written in the model in italics at the top left or right of the place. The type definitions are shown in Figure 5. They are similar to types in programming languages.

```
(**************** States*****************)
color Status = with SESSION|WAITINGRESV|RESVSENT|RESVRECVD;
color ParValues = with E|Ta|Tb|Fa|Fb;
color STSpec = subset ParValues with [E,Ta,Tb];
color SFSpec = subset ParValues with [E,Fa,Fb];
color State = product Status * STSpec * SFSpec;

(************RSVP Messages ************)
color TSpec = subset ParValues with [Ta,Tb];
color FSpec = subset ParValues with [Fa,Fb];
color DownStream = union sender:TSpec + resverror: FSpec;
color UpStream = union reserve: FSpec;
```

Figure 5: Colour set definitions.

The places Sender and Receiver have the type State. State is the product of the type Status, STSpec and SFSpec. Status is an enumeration type representing the four states (i.e. SESSION, WAITINGRESV, RESVSENT and RESVRECVD), which the service interface can have. SESSION is the initial state for both the sender and receiver. WAITINGRESV means that a sender request with the traffic characteristics of the data flow has been sent but no reservation request has yet been received. RESVSENT means that the receiver has sent a reservation request. RESVRECVD means that the sender has received a reservation request. STSpec and SFSpec are subsets of the type ParValues. ParValues is an enumeration type, which defines the values (including the empty value E) the parameters can have. STSpec represents the traffic characteristics of the data flow, which are stored as part of the state information. SFSpec represents the QoS characteristics of the data flow, which are also stored as part of the state information. For example, if the Receiver place contains the value (RESVSENT,Ta,Fb), it means that a reservation request, Fb, has been sent for the data flow with Ta traffic characteristics.The place SndToRcv has the type DownStream, which is the union of the types TSpec and FSpec and represents the information flow that travels downstream to the Receiver. The type TSpec is a subset of ParValues and defines the possible values of the data traffic characteristics. The type FSpec is a subset of the type ParValues and represents the value of the QoS characteristics included in the reservation request, which generated an error. The place RcvToSnd has the type UpStream, which is the union of the type FSpec and representsthe information flow that travels upstream to the Sender. The type FSpec represents the value of the requested QoS characteristics. It may be noted that the union type shows useful to represent both the downstream and upstream information flows because the selectors of the union colours sets can identify them. The *sender* and *resverror* selectors identify the downstream flows and the *reserve* selector identifies the upstream flow.

3.2.3Markings

Tokens are associated with each place. A token is a value (colour), which belongs to the type of the place. The marking of a place is the multi-set of tokens present on the place. It is a *multi-set*, since it may contain several tokens with the same value. For example, the place SenderUser may have the initial marking 2`Ta, which means that the place has two tokens, each with the value Ta. It means that the sender can send two requests with the same traffic values CPNs include the initial state of the system. It is called the *initial marking*. It is written on the upper left or right of the place. In the initial marking, each of the places Sender and Receiver has a single token with the value (SESSION,E,E), which means that no reservation nor traffic information has been sent yet (as indicated by the value E). Each of the places SenderUser and ReceiverUser has an initial marking consisting of two tokens 1`Ta++1`Tb and 1`Fa++1`Fb, respectively. It means that the sender user has two traffic requests with the values Ta and Tb, and the receiver user has two reservation requests with the values Fa and Fb. Initially, the remaining places do not contain any tokens.

3.2.4Transitions

Transitions represent the actions of the system. They are drawn as rectangles in Fig. 5. There are six transitions in the example. The transition RSVPSenderReq models the action taken when the sender sends a request with the traffic characteristics of the data flow. The reception and processing of a sender request is modelled by the transition RSVPSenderInd. The transition RSVPReserveReq models the action taken when the receiver generates a reservation request. The transition RSVPReserveInd is used to model the reception and processing of a reservation request. The transition RequestRejected models the action taken when the RSVP service provider fails in allocating the requested reservation. Finally, the transition RSVPResvErrorInd models the reception and processing of a reservation error notification.

3.2.5Arcs

Arcs connect transitions and places. A transition may have input places connected by incoming arcs and output places connected by outgoing arcs. Arcs have expressions associated with them. They are located next to arcs and determine which tokens are removed or added to the places as explained in the next section.

3.3. Dynamic Behaviour

The dynamic behaviour of the CPN system can be described as the marking of the net changing according to transition occurrences, which depend on the expressions of the surrounding arcs.

3.3.1Variables and Bindings

An arc expression is evaluated by assigning (*binding*) data values to variables. The result of the evaluation of an arc expression is a multi-set of tokens. The variable declaration of the example is shown in

Fig. 6.

```
(*************** Variables ***************)
var sta: Status;
var fspec,fspec1: SFSpec;
var tspec: STSpec;
```

Figure 6: Variable declaration.

A binding is represented in the form {v1=d1,v2=d2,…,vn=dn} where vi for i ∈ {1,2..n} is a variable and di is the data value assigned to vi. For example, assume that the places surrounding transition RSVPReserveReq have the current markings shown in Fig. 7, indicated next to the places. The token values are enclosed in boxes and the number of tokens
in small circles. For instance, the place Receiver has a current marking consisting of one token 1`(WAITINRESV,Ta,E). Two examples of bindings are:

b1={fspec=Fa,sta=WAITINGRESV,tspec=Ta,fspec1=E}
b2={fspec=Fb,sta=WAITINGRESV,tspec=Ta,fspec1=E}

Figure 7

For binding b1, the arc expressions associated with the two input arcs of the transition RSVPReserveReq evaluate to the multi-sets of tokens 1`(WAITINGRESV,Ta,E) and 1`Fa. For binding b2, the two input arc expressions evaluate to 1`(WAITINGRESV,Ta,E) and 1`Fb.A transition may have a boolean expression attached to it. It is called *guard* and is enclosed between brackets. Similarly to the arc expression, a guard may have variables. The guard hasto evaluate to true to accept the binding. Transition RSVPReserveReq has a guard attached to it. It specifies that the status of the receiver must be WAITINGRESV or RESVSENT to accept the binding.

3.3.2 Enabling and Occurrence of Transitions

A transition can occur if it is *enabled*. For a transition to be enabled in the current marking, it must be possible to *bind* (assign) data values to the variables appearing on the surrounding arc expressions and in the guard and the following conditions must be met. Firstly, each of the input arc expressions evaluates to tokens that are present on the corresponding input places. Secondly, if there is any guard, it must evaluate to true. The occurrence of a transition removes tokens from the input places and adds tokens to the output places. The removed tokens are the result of evaluating the expressions on the corresponding incoming arcs, while the values of the added tokens are the result of evaluating the arc expressions on the corresponding outgoing arcs. For example, the transition RSVPReserveReq is enabled in the current marking shown in Fig. 8 for the binding b1(introduced in the last section). The occurrence of transition RSVPReserveReq updates the marking of the place Receiver. Thus, the token representing the state of the receiver shows that the status of the receiver is RESVSENT and the QoS characteristics are Fa. It also adds a token representing a reserve (i.e. a reservation request), with the value of the requested QoS equal to Fa, to the place RcvToSnd. There can be more than one possible binding for the arc expressions surrounding a transition. These bindings define the way the transition can occur. However, for a given marking, the transition will be enabled for a subset of them. The execution of a CPN can be seen as an *occurrence sequence* consisting of markings that are reached and *steps*. A step consists of potentially several enabled binding elements that occur concurrently. A *binding element* includes a transition and a binding of its variables. For example, in the marking shown in Figure 8, there are three enabled binding elements as follows:

be1=(RSVPReserveReq,{fspec=Fb,sta=RESVSENT,tspec=Ta,fspec1=Fa})
be2 = (RequestRejected,{fspec=Fa})
be3=(RSVPReserveInd,{fspec=Fa,tspec=Ta,fspec1=E})

Figure 8

Binding elements be1 and be3 are *concurrently* enabled, since the multi-set of tokens resulting from the evaluation of the input arc expressions are present in the corresponding input places (i.e. the places Sender, RcvToSnd and Receiver) and the transitions use a disjoint sets of those tokens. Thus, these binding elements can occur together. It means that a request can be generated and sent by the receiver (transition RSVPReserveReq), while a previous reservation request is indicated to the sender (transition RSVPReserveInd). Binding elements be2 and be3 are in *conflict*, since they cannot get the only token (i.e. reserve(Fa)) in RcvToSnd place simultaneously. Therefore, the transitions RequestRejected and RSVPReserveInd need to share the token with each other. It means that a reservation request can be rejected (transition RequestRejected) or accepted (transition RSVPReserveInd). Binding elements be1 and be2 are also concurrently enabled.

4. CONCLUSION

This paper provides an introduction to the CPN used for modelling and analysis of RSVP. It gives an informal overview of CPNs by means of an example. Then it presents the CPN model of RSVP protocol, which is called Design/CPN. Finally, it describes the occurrence of RSVP protocol.

REFERENCES

[1] ISO/IEC, *Information Technology-Open Systems Interconnection- Guidelines for the application of Estelle, LOTOS and SDL*. TR 10167:1991, ISO/IEC, 1991.

[2] Jensen K. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*. Springer-Verlag, 2nd edition, *Vol. 1, April, 1997*.

[3] Jensen K. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use.* Springer-Verlag, 2nd edition, *Vol. 2, April, 1997*.

[4] Jensen K. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use.* Springer-Verlag, *Vol. 3, April, 1997*.

[5] Jensen K. (Ed). Special Section on *Coloured Petri Nets, International Journal on Software Tools for Technology Transfer, Springer, 1998*.

[6] Jezic G. *Formal Specification and Verification of the Multiparty Call in ATM UNI Protocol.* ConTEL 2009. Proceedings of the International Conference on Telecommunications and 2nd Broadband and Multimedia Workshop, University of Zagreb, Croatia, pp 307-314.

[7] Kelly D. *Automata and Formal Languages: An Introduction*. Prentice Hall, 2005.

[8] Kim Y. and Kim D. *Construction and Performance Measurement of the RSVP Local Network*. Proceedings of 1999 Internet Workshop (IWS99), IEEE, Osaka-Japan, February, 1999, pp 126-130.

[9] Kristensen L.M., Christensen S., and Jensen K. *The Practitioner's Guide to Coloured Petri Nets*. International Journal on Software Tools for Technology Transfer, Springer, 1998, *Vol. 2,* No 2, pp 98-132.

[10] Koustsofios E. and North S. *Drawing Graphs with Dot.* AT&T Labs-Research. March, 1999.

[11] Lindell B., Braden B., and Zhang L. *Resource Reservation Protocol (RSVP) -- Version 1:Message Processing Rules*. IETF Internet Draft, *February, 2008*.

[12] Macedonia M. and Brutzman D. *Mbone Provides Audio and Video Across the Internet*. IEEE Computer Magazine, *Vol 27,* No 4, *April, 1994,* pp 30-36.

[13] Meta Software Corporation. *Design/CPN Reference Manual for X-Windows*. Version 2, Meta Software Corporation, Cambridge, 1993.

[14] Milner R., Tofte M., Harver R. and McQueen D. *The Definition of Standard MLRevised*. MIT Press Books, 1997.

[15] Mohri M., Pereira F., and Riley M. *FSM Library*. AT&T Labs-Research. http://www.research.att.com/sw/tools/fsm/.

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

[17] Neogi A. and Chiueh T. *Performance Analysis of an RSVP-Capable Router*. IEEE Network Magazine, *September/October, 1999,* pp 56-63.