International Journal of Computer Science and Engineering (IJCSE)
Vol.1, Issue 1 Aug 2012 98-104
© IASET

MODELING A PROTOCOL USING COLOURED PETRI NETS

VEENA BHARTI[1]& SACHIN KUMAR[2]

[1]Assistant Professor MCA Department, RKGIT Ghaziabad, India

[2]Professor CS Department,AKGEC Ghaziabad, India

ABSTRACT

As a method of system modeling, Coloured Petri nets occupy an essential position in the fields of discrete events and dynamic systems. It is an important tool for concurrent system modeling and analysis. It is a combination model that can be represented by graph; it can express relations of concurrency, sequence, synchronization in system by a visual way. This paper analyzed simple protocol using Colored Petri net. Firstly, the definition of Colored Petri net was introduced, secondly, the working principle of simple protocol was described, and finally this paper model the simple protocol using coloured Petri nets.

KEYWORDS: Coloured Petri nets, CPNTool, Occurence Graph, State Table.

INTRODUCTION

Nowadays many new communication protocols have been created to improve the capability and the performance of networking technologies. It is important that the design of the protocol is proved to be free of significant errors to ensure that the protocol operates correctly without undesired or unsafe behavior. Formal methods are well-suited to protocol design activities [1]; they can increase confidence that the design is free of errors that would be expensive to fix once a protocol is deployed into the network. However due to the cost of applying formal methods (steep learning curve and time consuming), no upcoming standards apply them for protocol verification. This research aims to bridge this gap by automating the task of producing an executable formal model of common protocols.

Petri Net, is a formal method suitable to describe physical systems because of its ability to express concurrence and non-determinism. Besides the modeling, Petri net is suitable to verify systems employing a set of powerful methods based on strict mathematic analysis, for example state space, structure theory, and process algebra, etc. Colored Petri Net (CPN)[7] is a high-level Petri net, developed by Jensen Kurt at Aarhus University in Denmark in 1982. CPN models have graphical forms and a well-defined semantics which allows for formal analysis. CPN has been widely applied to a variety of applications [8], including network protocol, software engineering, artificial intelligence, operating system and data management.

In this paper, *CPNTOOLS*[5], a software package for modeling, simulation and analysis of CPN is used to model and analyze a protocol. The rest of this paper is structured as follows. In Section 2 we describe the assumptions and contribute a new definition of both state tables and CPNs that fits the assumptions. Section 3 shows the working of a simple protocol. Section 4 gives the model of a simple protocol by means of Occurrence Graph (OG). Finally, we will summarize our works in section 5.

COLOURED PETRI NETS

Formal definition of CP-nets Definition: A Coloured Petri Net is a tuple CPN = (S, P, T, A, N, C, G, E, I) satisfying the following requirements:

(i) S is a finite set of non-empty types, called colour sets.

(ii) P is a finite set of places.

(iii) T is a finite set of transitions.

(iv) A is a finite set of arcs such that:

• P □T = P □A = T □A = Ø.

(v) N is a node function. It is defined from A into P □T□T □P.

(vi) C is a colour function. It is defined from P into S.

(vii) G is a guard function. It is defined from T into expressions such that:

t □T: [Type(G(t)) = Bool □Type(Var(G(t))) □□]. (viii) E is an arc expression function. It is defined from A into expressions such that:

a □A: [Type(E(a)) = C(p(a))MS □Type(Var(E(a))) □□] where p(a) is the place of N(a).

(ix) I is an initialization function. It is defined from P into closed expressions such that:

p □P: [Type(I(p)) = C(p)MS].

Declarations

• Types, functions, operations and variables.

Each place has the following inscriptions:

• Name (for identification).

• Colour set (specifying the type of tokens which may reside on the place).

• Initial marking (multi-set of token colours).

Each transition has the following inscriptions:

• Name (for identification).

• Guard (boolean expression containing some of the variables).

Each arc has the following inscriptions:

• Arc expression (containing some of the variables).

When the arc expression is evaluated it yields a multi-set of token colours.

Enabling and Occurrence

A binding assigns a colour (i.e., a value) to each variable of a transition. A binding element is a pair (t,b) where t is a transition while b is a binding for the variables of t. Example: (T2,<x=p, i=2>).

A binding element is enabled if and only if:

• There are enough tokens (of the correct colours on each input-place).

• The guard evaluates to true. When a binding element is enabled it may occur:

• A multi-set of tokens is removed from each input-place.

• A multi-set of tokens is added to each output-place.

A binding element may occur concurrently to other binding elements – iff there are so many tokens that each binding element can get its "own share".

Main characteristics of CP-nets

Combination of text and graphics. Declarations and net inscriptions are specified by means of a formal language, e.g., a programming language.

• Types, functions, operations, variables and expressions. Net structure consists of places,

transitions and arcs (forming a bi-partite graph).

• To make a CP-net readable it is important to make a nice graphical layout.

• The graphical layout has no formal meaning. CP-nets have the same kind of concurrency properties as Place/Transition Nets.

There are many ways to model a protocol with CPNs, depending on the objectives of the modeler. In our work we consider the objectives of producing a CPN model that is structured similar to the state table description (for validation), and that is amenable to state space analysis (for verification). The modeling approach assumes a unicast protocol with two entities, sender and receiver, communicating by a single full-duplex channel. The current state of each entity, and associated state variables are stored in a single place. Each event is modeled by a single transition, when an event occurs the state's information is updated (containing the current state name ,next state name and state variables) For the communication channels we assume that there is no packet loss, and the timer events are considered nondeterministic (it either may occur or may not occur at any time).

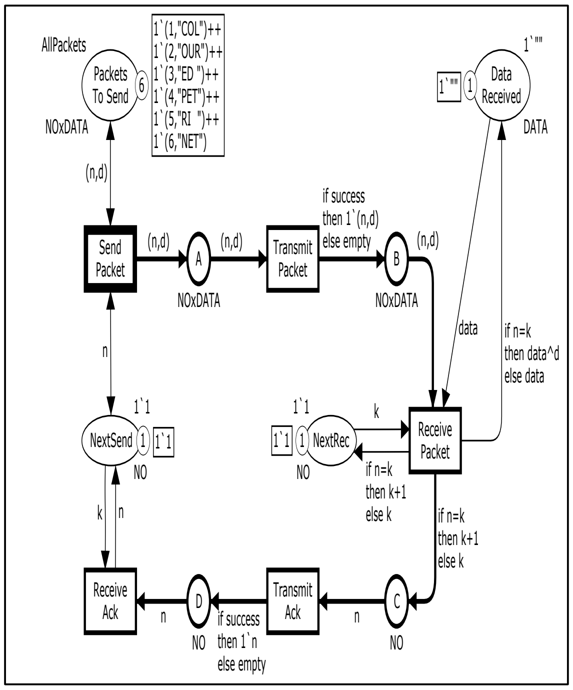
A SIMPLE PROTOCOL

Consider a simple protocol that can be used to transfer some information between a sender and a receiver and /or lose message. Therefore retransmissions are used to recover from message loss. A sender can be in one of three states: ready, waiting or done. They correspond to the sender: being ready to send data, awaiting from an acknowledgement from the receiver; or having received an acknowledgement and the data transmission having been accomplished.

There are two receiver states: ready: implying that receiver is ready to receive data; and processing: in which the receiver is busy with processing the data received.

Initially the sender is in its ready state, so it can send out the data, and then it transits to the waiting state, and starts a timer. The receiver is also in its ready state at the beginning. If the data arrives, the receiver receives the data, and moves to he processing state to process the data. After an acknowledgement is sent out, the receiver returns to the ready state. If the acknowledgement is sent to the sender, the sender will receive it, and change its state to done. However if the data or acknowledgement is lost in the channels or the procedure is delayed (either by the channel or by the receiver) the timer will expire before an acknowledgement is received. Then the sender goes to ready state again and it can retransmit the data as long as the retransmission counter has not reached its limit. So there can be more acknowledgements received by the sender. The protocol is designed in such a way that as long as sender has received one acknowledgement, it will go to the done state and consider the data has been successfully received by its peer, and when another acknowledgement comes back, it simply ignores the acknowledgement and does not change its state. A sender can also receive an acknowledgement when it is in ready state, which allows an acknowledgement to be accepted when the timer has expired but the data has not been retransmitted.

MODELING PROTOCOL WITH CPN



Sender Network Receiver

Simple Protocol

type INT = int;

type BOOL = bool;

type DATA = string;

type INTxDATA = product INT * DATA;

var n, k : INT;

var p,str : DATA;

val stop = "########";

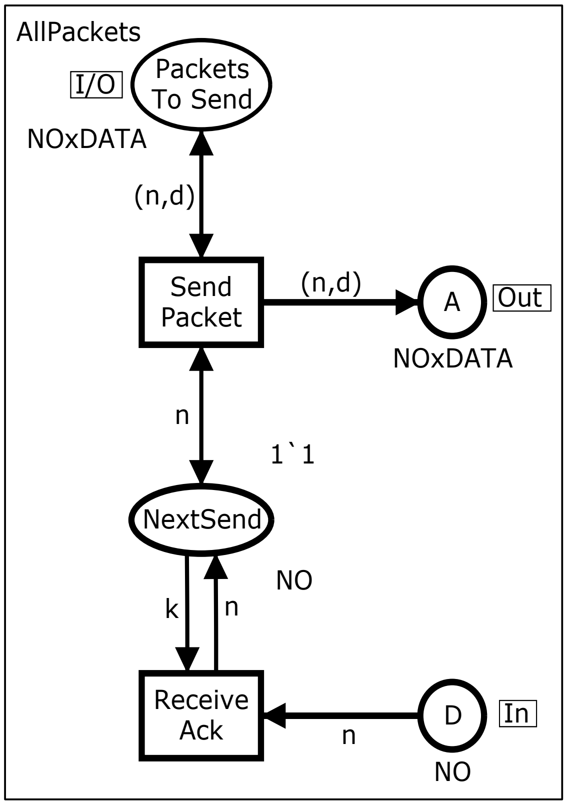
type Int_0_10 = int with 0..10;

type Int_1_10 = int with 1..10;

var s : Int_0_10;

var r : Int_1_10;

fun Ok(s : Int_0_10, r : Int_1_10) = (r□s);



Sender Module

Definition of colour sets:

colset NO = int; (* integers *)

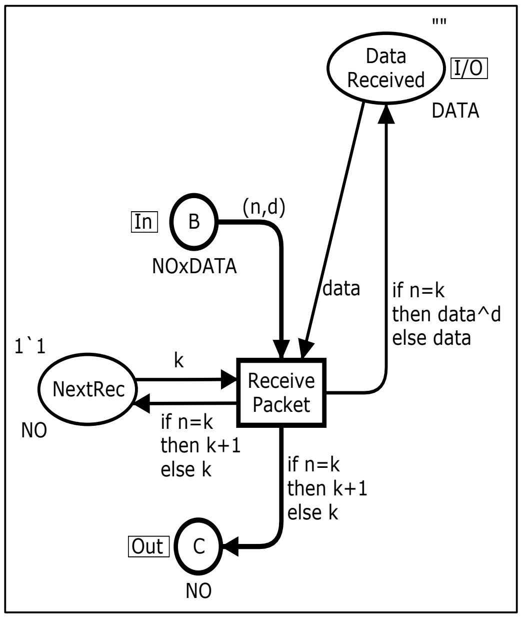
colset DATA = string; (* text strings *)

colset NOxDATA = product NO * DATA;

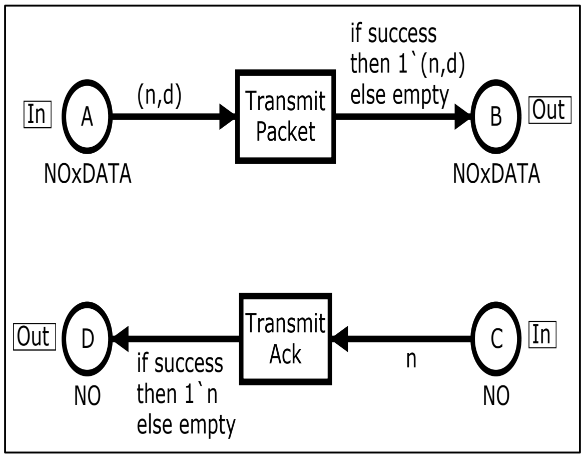
Each place may contain a number of tokens.

Each token carries a colour(data value).

The colour set of a place specifies the set of allowed token colours.



Receiver Module



Network Module

CONCLUSIONS AND FUTURE WORK

In this paper, we have modeled the protocol by means of CPN, standing out the most characteristic aspects of simple protocol. We have also analyzed the model through occurrence graph, which made it easy in the modeling formalization and easy to understand in computing. Colored Petri Net has the characteristics of good readability, expandability and graphic interface, it can control executive steps by simulation method [5].

REFERENCES

1. Kurt Jensen, “Colored Petri Nets: Basic Concepts, Analysis Methods and Practical Use”, Springer-Verlag, 1992.
2. B.Zouari, “High-level Petri net approach for supervisory control”, Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, v 2, 2003, p 1161-1166
3. Xiao Yuane,Kang Yingping. Design and Implementation of PPPoE Access Technology for Campus Network. Computer Knowledge and Technology.Vol.5,No.2S,September.2009.
4. Zhu Lianzhang,Sui Ruisheng,Kong Yingying. Simulation Based Performance Analysis In CPN Tools[J]. Micro-Computer Applications, 2008, 29(4).
5. S. Vanit-Anunchai, "Verification of railway interlocking tables using coloured petri nets;' in Proceedings of the 10th Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, Aarhus, Denmark, October 2009, pp. 139-158.
6. W. Stallings, Data and Computer Communications, 8th ed.Prentice Hall, September 2010.
7. K. Jensen and L. M. Kristensen, Coloured Petri Nets: Modelling and Validation of Concurrent Systems. Springer, 2009.
8. Luo Enta. Based on ethernet PPPoe protocol flow analysis and research. Journal of Hunan University of Science and Engineering. Aug.2008. Vo1.29 No.8.