Sindh Univ. Res. Jour. (Sci. Ser.), Vol: 37 (1) 47-56 (2005)

Timed Model for Protocol Verification and Performance Analysis

M. N. Brohi and B. S. Chowdhry*

Department of Information Technology, Preston University, Ajman, UAE.

Abstract

This paper presents an integrated approach to verify general properties of protocols and to analyze their performance based on a formal model called Integrated Timed Transmission Grammar (ITTG). It has been noted that un-timed protocol models cannot be used to verify a protocol in which time constraints are essential for the correct functioning of the protocol. ITTG is an extended and refined protocol model resulting from an evolutionary series of the Transmission Grammar-based models such as the Transmission Grammar (TG) and the Time Transmission Grammar (TTG). We can use our timed protocol model for verification of timed-dependent protocols as well as for performance analysis of protocols.

Keywords: Alternating Bit Protocol, Better Quasi Orderings, Rule Firing Time, Tempo-Blocking Cycle, Regular Grammar.

Introduction

In modern computer communication age protocols are playing vital role to exchange reliable and correct information. For this purpose we have to consider the correctness, robustness and performance of computer communication protocols. In this regard many UN timed and timed protocol models have been developed for protocol specification, validation and verification. For examples State-Transition Models (STM), Programming Language Models (PLM), Hybrid Models (HM), Communicating Finite-state Machines (CFSM), Untimed TG Model, Timed Petri Nets Models (PNM), Timed Calculus of Communicating Systems (CCS) Models, Timed Communicating Sequential Processes (CSP) Models, Timed Abstract Machine Models. It has been also noted that untimed models can not be used to verify a protocol in which time constraints are essential for the correct functioning of the protocol. In this connection in 1982, Shankar and Lam found that in order to prove a desired timeout condition for a simple protocol, UN timed modeling of that protocol is not adequate. A time pertinent model must be used to remove inherent limitations of the UN timed model, developed by Merlin and Farber, (1976).

Jain and Lam (1987) reported the necessity of timed protocol modeling when verifying a real-time protocol. Since a question was raised by Yemini and Kurose (1982) "Can current protocol verification techniques guarantee correctness?", so a functional correctness is not the only concern in protocol design. Another aspect of the protocol is its performance and as a matter of fact, the foundation should be once again timed models because without time specification performance analysis cannot be done in a formal model. In the same year they discovered that there is indeed a need to provide a unified approach to the functional and performance analysis of protocols. It is also interesting to note that most efforts in extending untimed models to timed models is for performance analysis of protocols. As it's a matter of fact that functional correctness is not the only concern in protocol design but another aspect of the protocol is its performance and the foundation should be the timed models because without time specification performance analysis cannot be done in a formal model. In front of timed protocol modeling there are two

* Department of Electronics & Telecommunication Engineering, Mehran University of Engineering & Technology, Jamshoro, Sindh Pakistan

major goals; first goal is for verification of timed-dependent protocol and second goal is for performance analysis of protocols. But very few models are targeted for both goals.

Present Timed Models

This section presents a brief description of some timed models; the Petri nets model, the petri nets and BQOs, the CSP model, the CCS, the CFSM model and Abstract Machine model.

Timed Petri Nets Models

Enormous work has been done on extending untimed Petri net models to timed models in order to model and analyze not only communication protocols, but also other systems, such as a real-time and multiprocessor systems. Nevertheless, our major concern here is those models related to protocols. Those models differ according to how time is associated with the net and in what form. Various researchers have used three different terms.

Timed Petri Nets and BQOs

This modelling approach has been developed by Abdullah and Nylen, (2001) in which they considered unbounded Timed Petri Nets (TPNs) where each token is equipped with a real valued clock representing the "age" of the token. Each are in the net is provided with a subinterval of the natural numbers, restricting the ages of the tokens traveling the arc. They applied a new method based on the theory of the **better quasi orderings (BQOs),** to derive an efficient constraint system for automatic verification of safety properties for TPNs.

Timed CSP Models

A timed CSP model was proposed by Reed and Roscoe (1986) to verify real-time properties of communicating processes while retaining compatibility with the semantics of the original UN timed model. It is an extension of Hoare's CSP trace model. Zic, (1987) extended the time CSP model by incorporating probability specification in the model. This is done by associating probabilities with CSP's non-deterministic choice operators. The purpose is to allow both protocol performance specification and verification in timed CSP.

Timed CCS Models

The timed model proposed by Nounou and Yemini (1984) is a timed CCS model even though they used a different set of notations. Basically, time information is not specified on the level of individual communicating entities, but on the level of the global behavior tree. The communicating entities are combined by parallel composition. The global behavior tree captures all the possible interaction sequences and non- deterministic behavior of a protocol.

Timed CFSM Models

Most work in this domain is done by researchers in the IBM Zurich Research Laboratory. Basically, there are three approaches to adding time specifications to the CFSM model. Two are done to predict performance of a protocol from its formal specification (Rudin, 1983, 1984; Kritzinger, 1984). The third one is done to verify a protocol modeled more realistically, namely by including time information of network components as part of the model.

Timed Abstract Machine Models

Shankar and Lam (1982, 1984) have proposed a timed abstract machine model that uses discrete valued timer variables to measure the lapse of time and time events to age the timer. Those time variables and time events are local to each process in the model. Timer variables from different processes are uncoupled and can tick at different rates. Nevertheless, an ideal timer is assumed, based on which local timers are constrained within a specified error bound by the accuracy axiom.

Introduction To Integrated Timed Transmission Grammar (ITTG) Model

All above described timed protocol models are extensions of the UN timed models. There are two major issues regarding the timed protocol modeling, so we have to consider both these issues in the light of the following points of view:

a. In what form the time specification is represented ?

b. With which component of the model is it associated ?

In case of first point of view, there are three types of time specification; Constant time, Time interval and Stochastic time and in lieu of other point of view there are two aspects of model components the time can be associated with model in global or model in local. In case of model in global, there are two methods to associate time specifications with the model. One methods is to associate time with the components of individual communicating entities before they are composed together e.g., time specifications in timed Petri nets and timed abstract machines and the other method is to associate time with the components of the global protocol behaviour after individual entities are composed together e.g., time specifications in timed CCS. In case of model in local, the methods to associate time specifications are totally model dependent except some natural cases As in case of Petri net, there are three main required components: state (Place), transition, and arc. Similarly, time can be associated with either states **(e.g., Moore Machine, 1956)** or transitions **(e.g., Mealy Machine, 1955)** in the CFSM model, with events in CCS and CSP and with state variables in an abstract machine.

Analytic power of the timed models can also be discussed from two points of views:

(A) What purpose is it used For, verification or performance?

(B) How is it related to its original untimed model?

In relation to above points, we have following observations:

Form of time specification

Stochastic time specification is only suited for performance analysis.
Time interval specification is best suited for verification, but its analytic method is hard to derive.

Constant time specification can be used for verification and performance analysis. But when used for verification, it is not as good as time interval specification, when used for performance analysis; it is not as good as stochastic time specification.

Moreover, probability specifications need always be brought into the model, if performance analysis is going to be supported by non-stochastic time specifications such as constants or intervals.

Time association with the model components

We believe that the association of time specification with individual communicating entities is more realistic and more convenient than with the global behavior of the protocol.

How time is associated with the components of communicating entities, strongly decides how difficult it can be to derive an analytical method. Thus it should be done with the analytic method in mind.

III. Which untimed model is better for time extension?

It seems that time extension in one method that can often be applied to another model. Thus the one on which untimed model the time extension is done, is not as important as other issues. One of these issues is the lack of a timed model that can support both verification and performance analysis of protocols.

Here, first of all we are describing previously presented timed protocol model i.e., the TTG based on grammar and then we are presenting our developed timed protocol model based on this model. The TTG model is an extension of the UN timed TG model for the verification of time-dependent and time-independent, synchronous and asynchronous protocols.

The extension was done on the following basis:

The specification is divided into two classes, entities and channels.

Time specifications are added to the model. There are three types of time information that can be used in the model i.e., rule firing time, transmission delay and timeout interval. The rule firing time is associated with each production rule of the entities to show the delay and execution time of the rule. The transmission delay is associated with each production rule of the channels to show how long a message will take to reach its destination. The timeout interval is specified when a timer is activated.

For activating and deactivating a timer there are two actions known as set timer and clear timer. There is a timeout handler for individual timer in an entity, which specifies the service actions taken when the timer expires.

Each time specification in the TTG model is in the form of an internal $[t_{min}, t_{max}]$, where t_{min} is a nonnegative integer, t_{max} is a nonnegative or infinity and $t_{max} \geq t_{min}$ When specified as a rule firing time, t_{min} is the minimum delay time before the actions in a production rule can be executed after the rule is enabled and $t_{max}$is the maximum elapsed time before which all the actions in an enabled rule must be completed. When specified as a timeout interval or as a transmission delay, t_{min} and t_{max} are the minimum and maximum times required for a timer to expire and for a message to reach its destination, respectively.

The TTG model is capable of handling time-dependent protocols; it can be applied to only a restricted class of protocols. To overcome this drawback, we have modified the TTG model and developed a novel algorithm of timed reachability analysis based on the new model. In order to distinguish the new model from the old TTG model it is called the ITTG model. The ITTG is the modification of the TTG model.

3.1 Reachability Analysis

It is a global state exploration process that starts from the current (initial) global state and recursively explores all the possible transitions that lead to new global states.

The result is a reachability graph, which captures all possible states.

ITTG Model

ITTG is an integrated model to verify the general properties of protocols and to analyze their performance based on a formal model. It is an extended and refined protocol model resulting from an evolutionary series of the transmission grammar - based models. In fact the extension on the TG-based models follows the same line of evolution, as other models like from UN timed to timed modeling of protocols. But we are integrating two major goals of timed protocol models, viz, verification and performance analysis in a single framework. The major extension done to previous timed model (TTG) is the incorporation of time specifications into the model. Now in order to facilitate performance analysis also both time and probability specifications are incorporated into the ITTG model.

The ITTG model is a set of regular grammars. It consists of three sub regular grammars or set of regular grammars; **entities, channels and timeout handler**. In ITTG, there are **terminals and non-terminals** known as **actions and states.**

Entities in the ITTG Model

An entity is a regular grammar preceded by "**Ts**. entity entity-id". The actions performed by an entity are given below:

Ms. entity-id[.entity-id…].message-name

Mr. entity-id. message-name

Ts. timer-id. time-interval

Tr. timer-id internal-action-name

Where **Ms** and **Mr** specify communicating actions corresponding to sending and receiving of a message, similarly there are two timer actions denoted by **Ts** and **Tr** known as set timer and reset timer to identify the actions taken by timer and internal-action specifies those operations invisible to other entities.

Each production rule of an entity is given below:

<Current-state>.[time-internal]::=[probability-value][action-1,action-2,…<next-state>].

Where action-1 is either a **Ms, Mr** or an internal action, but any action that follows must be a timer action. The <current-state> that appears in the first production rule of the grammar is implicitly defined as the initial state of the entity. Usually, a state may have more than one production rule. In ITTG, these rules are grouped together and each of them is separated from others by a comma (","). A state is said to be **passive** if none of its production rules contains a **Ms** or an internal action otherwise it is **active**. For an active state, a time interval must be specified and associated with it. Semantically, this time specification gives the minimum and maximum delays that an active state must be held before it can move to next state. Conversely, for a passive state no time interval need be specified because how long a passive state is held will be decided by other external events. An active state with more than one production rule containing a **Ms** or an internal action is known as a decision state. For a decision state a probability value must be specified for each of its production rules containing either a **Ms** or an internal action, such that the sum of all the probability values assigned to these rules is 1.

Channels in the ITTG Model

A channel in ITTG is a regular grammar preceded by "**Ts**. Channel [<entity-id-1> entity-id-2] of size number", which

specifies not only the source and destination of the medium but also the capacity of the medium. A channel has only one state known as the idle state and its possible actions are given below:

MC. in-message-name

ML. in-message-name

MD. in-message-name

MG. in-message-name. out-message-name

Where **MC, ML MD** and **MG** are denoted by correct message transmission through the medium, message loss in the medium, message duplication by the medium and message in-message-name has been garbled during the transmission and the garbled message delivered by the medium is out-message-name respectively. In other words we can say that these actions specify possible behavior of a channel. For convenience of specification, if a medium has no discrimination against the message type, message name **"*"** is used to indicate any message sent through the medium.

The production rule of a channel is given below:

<idle-state>::=[probability-value][time-internal]action<idle-state>

Where action is either a **MC, ML, MD** or **MG** and time-interval specifies how long it will take for a message to get through the medium. For accurate estimated channel busy time it is necessary to specify a time interval even for the lost message in the medium.

A channel with multiple production rules for an in-message-name is known as an unreliable channel. For this type of channel a probability value must be specified for each alternative production rule of the in-message-name such that the sum of the probability values assigned to these rules is 1.

Timeout handler in the ITTG Model.

A timeout handler is a set of regular grammars preceded by "**Ts**. timeout-handler timer-id of entity entity-id for (message-name, acknowledge-name) in channel entity-id = entity-id, [(message-name, acknowledge-name) in channel entity-id = entity-id...]", where both the entity and the message (acknowledgements) served by a particular timer are specified. Each grammar in the set consists of only one production rule. Each production rule of a timeout handler refers to the state of the entity that the handler serves rather than the state of the handler itself; as such it carries different semantics. The form it takes is as follows:

<Current-state>:: = time-internal action-1 [action-2...]<next-state>.

Where <Current-state> must be unique for each timeout handler and a time-interval must be specified to indicate the time taken execute this timeout service. The actions that can be performed by a timeout handler are **Ms**. entity-id [.entity-id,].message-name and **Ts**. timer-id.time-interval, which model the timeout retransmission mechanism normally employed in communication protocols. Semantically, the rule specifies what kind of service should be done, when a timeout occurs due to certain timer and in a certain state of the entity.

Here we are considering the alternating bit protocol (ABP), which is a very famous protocol for communication between two parties over an unreliable medium. ABP is a connection-less protocol for transferring messages in one direction between a pair of protocol entities. Each message that is sent over the medium has a single bit added, either a 1 or a 0. This bit can be

used to determine if this message is a duplicate. The ABP described here takes into consideration the fact that the medium may lose a message in transit.

Fig.1 shows the ITTG state model of the alternating bit protocol using the reachability analysis and **Fig.2** lists the formal specification of the protocol in ITTG model. After performing the reach ability analysis, the ABP is found to be free from all erroneous protocol properties such as unspecified reception, unspecified timeout service state, deadlock, channel overflow, improper timer action and premature timeout. Nevertheless, three tempo-blocking cycles are identified:

29→27→28→29, 9→23→13→14→8→9 and 14→16→17→14

Once logical correctness of the protocol is verified, the next step is to compute performance measures of the protocol based on the global state graph already available after the verification. First, the timed probabilistic graph in the best and the worst throughput cases are extracted. Then, based on the extracted timed probabilistic graphs we get the following performance measures after computation.

Channel Utilization

1→2: [0.291971, 0.299774] or approximately 30%

2→1: [0.294102, 0.301963] or approximately 30%

Throughput: [0.021607, 0.044370] or 21.61, 44.37 (If one time unit = 1msec.)

Efficiency: [0.571880, 0.634422] or approximately 60%

Basically, the ABP specified here can transfer from 21.7 up to 44.4 messages per second. Both channel utilizations are approximately 30% without much difference under the best and the worst cases and about 60% of the time the protocol is doing something effective.

Below given Fig.2 lists the formal specification of the protocol in ITTG.

Tsentity 1.

Current State	**Next State**	
<1>. [0,10]::=SE<2>.		
<2>. [1,1]::=Ms.2.DF0, Ts.Timer. [25,30]		<3>.
<3>	::=Mr.2.Er, Tr.Timer	<2>.
	Mr.2.AK1, Tr.Timer	<2>.
	Mr.2.AK0, Tr.Timer	<4>.
<4>.[0,10]::=SE<5>.		
<5>. [1,1]::=Ms.2.DF1, Ts.Timer. [25,30]		<6>.
<6>	::=Mr.2.Er, Tr.Timer	<5>.
	Mr.2.AK0, Tr.Timer	<5>.
	Mr. 2.AK1, Tr.Timer	<1>.

Tstimeout-handler Timer of entity 1 for (DF0, AK0)

in 1=2, (DF1, AK1) in 1=2.

<3> ::=[1,1] Ms.2.DF0, Ts.Timer.[25,30]<2>.

<6> ::=[1,1] Ms.2.DF1, Ts.Timer[25,30]<5>.

Tsentity 2.

<1> ::=Mr.1.DF0 <2>.

Mr.1.Er <6>.

Mr.1.DF1 <6>.

<2>.[1,1] ::=RE <3>.

<3>. [1,1]::=Ms.1.AK0 <4>.

<4> ::=Mr.1.Er <3>.

Mr.1.DF0 <3>.

Mr.1.DF1 <5>.

<5>.[1,1] ::=RE <6>.

<6>. [1,1]::=Ms.1.AK1 <1>.

Tschannel 1→2

<Idle> ::=0.8[5,10]MC.*.

::=0.1[5,10]ML.*.

::=0.1[5,10]MG.*.Er.

Tschannel 2→1

<Idle> ::=0.8[5,10]MC.*.

::=0.1[5,10]ML.*.

::=0.1[5,10]MG.*.Er.

Fig. 2. ITTG Model Based On Alternating Bit Protocol

Where in Fig.1 and Fig.2, the labels **SE** and **RE** stand for two service primitives (send and receive, respectively) provided to the user at the higher layer. Event **SE** receives a data frame from one user on the sender site, while event **RE** delivers the received frame to the user on the other user on the receiver site. There are two types of data frames (**DF0** and **DF1**) and two types of acknowledgement frames (**AK0** and **AK1**), **0** and **1** represent values of the control bit. Transmission errors are shown as **Er**. In Fig.2, from L.H.S <> is a current (initial) state and R.H.S <> is a final state, [] is a time interval, **0.8, 0.1** is probability value, **Mc**, **ML**, **MG** are

actions and L.H.S <2> and <3> are active and passive states respectively.

Conclusion.

The ITTG timed model has been developed for both protocol verification and performance analysis. Basically, verification of a protocol is done based on the properties of both reachable states and their reachability graph. On the other hand, performance analysis of a protocol is done based on the extraction of timed probabilistic graphs from the global reachability graph. The final measures of protocol performance are represented in the form of an interval, indicating the performance parameters of the protocol under the best and worst cases. The utilization of channel 1 2 and 2 1 are approximately 30% without much difference under the best and the worst cases and about 60% of the time the protocol is doing something effective.

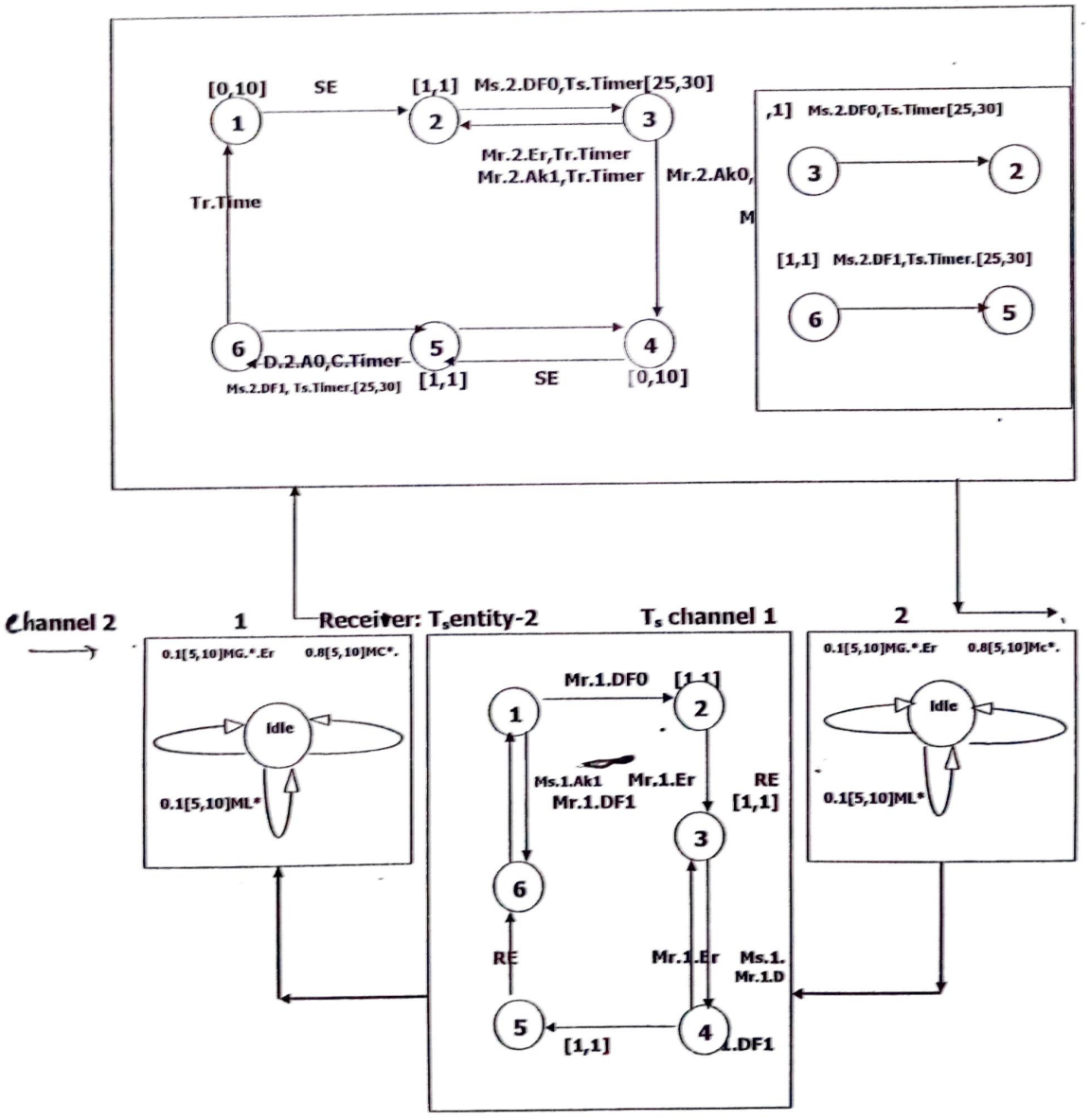

Fig.1. ITTG State Model Based On Alternating Bit Protocol

References

Jain, P. and S. S Lam (1987) Modelling and verification of real-time protocols for broadcast networks. IEEE Trans. Software Engineering **SE-13** (8), 924-937.

Kritzinger, P.S. (1984). Analyzing the time efficiency of communication protocol. Proc. IFIP Int. Workshop on Protocol Specification, Testing and Verification. 4th, pp.527-540.

Mealy, G. H. (1955). A method for synthesizing sequential circuits, Bell System Technical J.**34 (5):** 1045-1079.

Moore,E.F. (1956). Gedanken experiments on sequential machines, Automata Studies, pp.129-153, Princeton Univ. Prees, Princeton, NJ.

Merlin, P. M. (1976). A methodology for the design and implementation of communication Protocols. IEEE Trans. Comm. **COM-24** (6), 614-621.

Nouno, N. and Y. Yemini, (1984) Algebraic Specification - Based Performance Analysis of Communication Protocols. PSTV **1984:** 541-560.

Parosh, A. A and A. Nylen, (2001) Timed Petri nets and BQOs. In Proc. ICATPN 22nd Int. Conf. on application and theory of Petri net, volume 2075 of Lecture Notes in Computer Science, pages 53-70.

Reed, G. M and (1986) Roscoe A. W. A Timed Model for communicating Sequential Processes. ICALP **1986** 314 -323.

Rudin, H. (1983) from formal protocol specification towards automated performance Prediction. Proc. IFIP Int. Workshop on Protocol Specification, Testing and Verification. 3rd, pp.257-272.

Shankar, A. U. and S. S. Lam, (1982) On time-dependent communication protocol and their Projections. Proc. IFIP Int. Workshop on Protocol Specification, Testing and Verification; 2nd, pp.215-236.

Yemini, Y. and J. F. Kurose, (1982) can current protocol verification techniques GuaranteeCorrectness, computer Networks **6** (6): 377-381.

Yemini, Y. and J. F. Kurose, (1982) Towards the unification of the functional and Performance analysis of protocols, PSTV: 182-196.

Zic, J. J. (1999) Extensions of Communicating Sequential Processes to allow protocol Performance specifications. Computer Communication Review **17** (5): 217-227.