Timed Protocol Verification for Estelle-Specified Protocols*

Chung-Ming Huang and Shiun-Wei Lee

Laboratory of Computer Aided Protocol Engineering (LOCAPE)
Institute of Information Engineering
National Cheng Kung University
Tainan, Taiwan 70101
R.O.C.

Correspondence: huangcm@locust.iie.ncku.edu.tw

Abstract

This paper presents a new model, which is named Timed Communicating State Machine (TCSM), for specifying protocols that incorporate timed properties as part of their specifications. The TCSM model is similar to the Extended Communicating Finite State Machine (ECFSM) model. The major extension is providing an additional mechanism, i.e., a time interval attribute, for describing the timed properties. We also propose the corresponding formal TCSM-based timed verification scheme, i.e., a new timed global state reachability analysis. In the new timed global state reachability analysis, the exploration is decided by the predicate, the time relationships, and/or the input event. Using the TCSM model and the timed verification scheme, an Estelle-based Timed Protocol Verification System (ETPVS) has been developed on SUN SPARC workstations. In this way, timed protocols can be formally specified in Estelle and can also be verified using ETPVS.

Keywords: Protocol Engineering, Timed Protocol Verification, Estelle, Formal Modeling.

---

*The research is supported by the National Science Council of the Republic of China under the grant NSC 83-0408-E-006-013.

1 Introduction

Formal methods have been widely applied to specify communication protocols and to analyze their properties [27]. These models include Communicating Finite State Machines (CFSM) [6, 30], Petri Net [10], etc. These models, however, can't precisely specify protocols with timed properties. With the awareness of inadequacy of formal models without time specifications, many timed models, including those proposed in the Protocol Engineering field [5, 25, 26, 35, 32, 33] and those proposed in the real-time systems field [14, 15, 34], have been used to meet the requirements of specifying timed properties.

In the past decade, Formal Description Techniques (FDTs) have been proposed for formally specifying communication protocols. These FDTs include ISO's Estelle [7] and LOTOS [4], and CCITT's SDL [3]. However, there are not many FDT-based methodologies and tools that are devoted to formally verify timed properties. One of the major reasons is that currently existing formal models, e.g., those models mentioned above, for specifying timed properties are not well matched with FDTs' specifications.

Realistic communication protocols always have timed properties because data transmission, data processing, and the occurrence of errors need time to process and reflect. Additionally, the two ends of communication execute in parallel and some of their operations have timing constraints for correct system behaviors. Therefore, a formal model should be able to express concurrency, communication, synchronization, and time explicitly. A corresponding verification scheme is also required to analyze the embedded timed properties of the specified protocols. To facilitate automatic verification, executable verification schemes should be easily derived based on the formal models. A verification scheme needs to be executable so that formal protocol specifications can be simulated or fast-prototyped. The reason is that in practice there are many properties that are not feasible to express or prove formally, but must be observed experimentally.

In [25], Lin and Liu have proposed an integrated approach for timed protocol verification and performance analysis. However, Lin and Liu's model does not allow the existence of predicate. Therefore, Lin and Liu's model cannot be straightforwardly applied to state-transition-based FDTs, e.g., Estelle. Under this concern, we propose a novel model, which is called Timed Communicating State Machine (TCSM), for formally specifying protocols with timed properties. We select state machines because of the proven convenience of the notation. Futher, it is easy to trace and analyze protocols automatically, and it is well-matched with Estelle's notations using the state machines approach. The main difference between our method and Lin and Liu's method are as follows: (1) The TCSM model allows the existence of predicates in transitions; (2) The TCSM model is directly matched with ISO's Estelle. In contrast, Lin and Liu's model is not so straightforward to be applied to ISO's Estelle.

The TCSM model is similar to the Extended Communicating Finite State Machine (ECFSM) model [27]. The major difference between the TCSM model and the ECFSM model is the incorporation of time specifications into our TCSM model. The time specification is integrated in the condition part of the transition in the form of a time interval clause. The interval specification is restricted to non-negative integers only. In this way, a transition can be time-bound by an interval that defines the smallest time (delay) and the largest time (deadline) for making a transition. Using the TCSM model, each communicating entity is specified as a TCSM and protocol entities can communicate with each other via the one to one unidirectional channels. Each channel is also specified as a TCSM that can represent various conditions, e.g., messages may be transmitted successfully or may be lost.

To analyze the timed properties of the protocols specified in the TCSM model, a verification scheme (algorithm) is sketched in this paper. The scheme is basically a new global state reachability analysis for the TCSM model. In the new global state reachability analysis, each global state consists of a global state matrix, which records the status of each TCSM and each communication channel, and a temporal precedence matrix, which records the relative time relation between each pair of occurrable events. Using the TCSM model and the proposed timed verification scheme, an Estelle-based Timed Protocol Verification System (ETPVS) has been developed on SUN SPARC workstations.

The rest of this paper is organized as follows. In Section 2, the TCSM model and some definitions are introduced. In Section 3, the verification scheme that is applicable to the TCSM model is presented. An example of using the scheme is described. In Section 4, the Estelle-based Timed Protocol Verification System (ETPVS) is introduced. In Section 5, some discussions and conclusion remarks are given.

2 The Timed Communicating State Machine (TCSM) Model

In this section, the Timed Communicating State Machine (TCSM) model is presented. Some definitions and notations for protocol specifications using the TCSM model are introduced.

Definition 2.1: Each communicating entity or channel of a protocol is specified by a Timed Communicating State Machine (TCSM). Each TCSM is represented as an eight-tuple $(\Sigma, S, s_0, V, T, P, A, \delta)$, where

1. Σ is the set of messages that can be sent or received,
2. S is the set of states,
3. s_0 is the initial state,
4. V is the set of context variables,
5. T is the set of time intervals associated with transitions,
6. P is the set of boolean expressions that operate on context variables,
7. A is the set of actions that operate on context variables,
8. δ is the set of state transition functions, where a state transition can be formally represented as follows: $S \times \Sigma \times P(V) \times T \rightarrow \Sigma \times A(V) \times S$.

Figure 1: A generic state transition in the TCSM model

Figure 1 shows a TCSM-based state transition, where a circle represents a state, an arc represents a transition, ”?*ent·mess*” represents that message *mess* is input from entity *ent*, and ”!*ent·mess*” represents message *mess* is output to entity *ent*. For convenience, a state transition is represented as S_h-T $\rightarrow S_t$, where T is called a transtion, $S_h(S_t)$ is called the head (tail) state of T, and T is called an incoming (outgoing) transition of $S_t(S_n)$. A transition has two parts: (1) the condition part, and (2) the action part. Depending on the transition type, the condition part may consist of an input event, a predicate and/or a time clause, which is denoted by delay$[t_{min}, t_{max}]$; the action part may consist of a number of statements that operate on context variables, and/or output events. A time interval $[t_{min}, t_{max}]$ in the delay$[t_{min}, t_{max}]$ clause indicates the minimal time (t_{min}) that the transition must be delayed, and the maximal time (t_{max}) indicates the deadline that the transition must occur, at the transition's head state. That is, let T denote the time the corresponding TCSM enter into the head state of transition E. Transition E can be executed after T+t_{min}, and should be executed before T+ t_{max}. If a transition has a predicate, then the predicate must be true for the transition to be selected and executed.

In the TCSM model, there are four transition types: spontaneous transitions, when transitions, channel transitions, and timer transitions. Except when transitions, these transitions may have time intervals, which represent the (state) holding time, transmission time, and time-out time respectively.

Definition 2.2: A *spontaneous transition* of a communicating entity is a transition that has no input event, but has output events. The corresponding state transition function is $S \times T \times P(V) \rightarrow \Sigma \times A(V) \times S$.

Figure 2-(a) shows an example of a spontaneous transition.

Definition 2.3: A *when transition* of a communicating entity is a transition that has an input event with/without output events. The corresponding state transition function is $S \times \Sigma \times P(V) \rightarrow \Sigma \times A(V) \times S$.

Figure 2-(b) shows an example of a when transition.

Definition 2.4: A transition in a channel entity, which is named as a *channel transition*, has an input event with/without an output event. The corresponding state transition function is $S \times \Sigma \times T \rightarrow \Sigma \times S$.

The transitions in channel entities are different from when transitions and spontaneous transitions. In the TCSM model, a channel entity acts as a message transformer. A transition in a

(a) (b)

(c) (d)

Figure 2: (a) A spontaneous transition, (b) a when transition, (c) a channel transition, (d) a timer transition

channel entity has an input event, but may have or not have an output event. A channel transition without an output event represents the input message is lost in the channel. A transition with an output event indicates successful transmission if the output message is the same as the input message, indicates a garbled transmission if the output message is different from the input message, and indicates other conditions depending on specifications. Figure 2-(c) shows an example of a channel transition.

Definition 2.5: A *timer transition* is associated with a delay clause without the predicate and without input/output events. The corresponding state transition function is $S \times T \rightarrow S$.

Figure 2-(d) shows an example of a timer transition.

Spontaneous transitions, channel transitions, and timer transitions are also called active transitions. Each active transition is associated with a time interval delay [t_{min},t_{max}]. When the associated predicate is true, an active transition can be executed at any instant in $[T+t_{min}, T+t_{max}]$, where T is the time the corresponding TCSM enters into the head state of the active transition. A when transition is called a passive transition. There is no time specification in a passive transition. Each passive transition can be executed when the input message is available and the associated predicate is true.

In order to realize the possible behaviors of protocols, all of the possible transition sequences exist among protocol entities are explored. The exploration is called global state reachability analysis [27]. In the TCSM model, a global state representation structure is associated with a time specification. Using an approach similar to that in [25], the global state structure for the TCSM model contains two matrices, a global state matrix and a temporal precedence matrix.

Definition 2.6: A *global state matrix* is a matrix whose diagonal entries represent the status

(a) (b)

Figure 3: The global state structure, (a) the global state matrix part, (b) the temporal precedence matrix part.

of the communicating entities, and whose non-diagonal entries record existing messages of channel entities. Each diagonal entry is in fact a state vector that includes the TCSM's state, and context variables' values. Each non-diagonal entry includes the messages in the communication channel.

Figure 3-(a) shows the state matrix of a global state. Given a global state GS, each entity's outgoing active transitions in GS are called occurrable events of GS. Depending on the specification, each occurrable event E is associated with some occurrence time bounds, which identifies when E can be executed. Thus, when an occurrable event is selected to occur, the other occurrable events' occurrence time bounds are adjusted accordingly. Therefore, each occurrable spontaneous transition, channel transition, or timer transition event is associated with an occurence time bound, which is called remaining time. The initial remaining time of each occurrable event is the $[\mathrm{t}_{min},\mathrm{t}_{max}]$ that is specified in the corresponding delay clause. When the remaining time expires, the occurrable event can be fired and the whole system is changed to the next global state. The occurrence of an occurrable event may generate some new occurrence events, and may remove some of previously occurrable events. Additionally, the remaining time of each occurrable event that is not removed needs to be re-calculated. Thus, each global state matrix is associated with an occurrable event list, which contains all occurrable events of each entity and each event is associated with its remaining time.

Definition 2.7: A *temporal precedence matrix* maintains the relative time relationships between each pair of occurrable events. Each entry is derived when the paired occurrable events co-exist at the first time in the global state reachability analysis.

Figure 3-(b) shows the temporal precedence matrix of a global state.

A global state transition is represented as $GS \overset{E}{\rightarrow} GS'$, where E is called a global transition, GS (GS') is the parent (child) global state of GS' (GS), and E is also called the outgoing (incoming) global transition of GS (GS'). A global transition can consist of one or more (entity) transitions. Depending on the executed active (entity) transitions, global transition E has the following three

Figure 4: The occurrence of a type S global transition.

types:

1. Type S: A communicating entity executes a spontaneous transition that contains one/many output event(s). There must be one/many peered channel transition(s) to transform the message(s). These channel transitions can be a successful transmission transition, a message-lost transition, a message-garbled transition, etc., depending on the specifications.

   An example is shown in Figure 4. In communicating entity a, there is a spontaneous transition T_1 that has two output events, whose peered channel entities are b and c. In the peered channel entity b, there are three transitions, i.e., T_2, T_3, and T_4, that can transform the corresponding message sent from communicating entity a. T_2 is a successful transmission transition, T_3 is a message-lost transition, and T_4 is a message-garbled transition. In the peered channel entity c, there are two transitions, i.e., T_5 and T_6, that can transform the corresponding message sent from communicating entity a. T_5 is a successful transmission transition, and T_6 is a message-lost transition. Thus, there are six types of global transitions accordingly: (1) two channel transitions are both successful transmission transitions, (2) a successful transmission transition in channel entity b and a message-lost transition in channel entity c, (3) a message-lost transition in channel entity b and a successful transmission transition in channel entity c, (4) two channel transitions are both message-lost transitions, (5) a message-garbled transition in channel entity b and a successful transmission transition in channel entity it c, (6) a message-garbled transition in channel entity b and a message-lost transmission transition in channel c.

2. Type C: A channel entity executes a channel transition. According to the peered ”when” transition, which receives the message in the communication channel, there are two subtypes: type C-I is the peered ”when” transition has no output event, and type C-II is the peered ”when” transition has some output events.

   An example is shown in Figure 5. Let channel entity a have a channel transition T_1 that outputs message m to communicating entity b, which is depicted in Figure 5-(a). Figure 5-(b) shows an example of type C-I, i.e., communicating entity b has a ”when” transition T_2, which can receive message m and has no output event in the action part. In this case, there is one global transition, which is also depicted in Figure 5-(b). Figure 5-(c) shows an example of type C-II, i.e., communicating entity b has a ”when” transition T_3, which can receive m

(a) (b)

(c)

Figure 5: The occurrence of a type C global transition event.

and can output message n to channel entity c. In b's peered channel entity c, there are three transitions, i.e., T_4, T_5, and T_6, that can transform message n. T_4 is a successful transmission transition, T_5 is a message-lost transition, and T_6 is a message-garbled transition. In this case, there are three global transitions: one global transition contains T_4, another global transition contains T_5, and the other global transition contains T_6.

3. Type T: A communicating entity executes a timer transition. A timer transition does not communicate with other entities. Thus, the global transition contains only the timer transition.

3 Timed Global State Reachability Analysis for TCSM

In this section, the timed global state reachability analysis for the TCSM model is presented. The logical errors that can be detected are defined at first. Next, the time interval addition and subtraction is introduced. Then, the derivation of temporal precedence matrices and the generation of child global states from a global state are described in detail. Finally, the timed global state reachability analysis algorithm is presented.

3.1 Logical Errors

In the TCSM-based timed global state reachability analysis, some logical errors can be detected. These errors are defined as follows:

Definition 3.1: A global state contains a *deadlock error* if the following conditions are satisfied: (1) all communication channels are empty, and (2) the global state has no occurrable event.

Definition 3.2: A global state contains an *unspecified reception error* if one of the following conditions is satisfied: A message m in a channel C is due according to the time specification of m, but the entity B, which should input messages from C, either (1) has no specified "when" transition that can receive message m at B's current state; or (2) has the corresponding "when" transition that is specified to receive message m at B's current state, but the associated predicate is false.

Definition 3.3: A global state contains a *channel overflow error* if the number of messages in a communication channel is greater than the channel size.

Definition 3.4: A global state contains a *premature time-out error* if a timer expires without any loss of the message or without any loss of the response of the corresponding message that the timer is activated for.

Definition 3.5: If the communication is restricted to be First-In-First-Out (FIFO), a global state contains a *non-FIFO error* when a lately generated message in a communication channel A can be received earlier than more early generated messages in A[1].

Definition 3.6: A global state contains a *transmitted lock error* if the following conditions are satisfied: (1) all of the outgoing transitions of a communicating entity are spontaneous transitions, and (2) the associated predicates are all false.

3.2 Time Interval Addition and Subtraction

Time interval addition/subtraction are different from the normal numerical addition/subtraction. Let the time interval of event C be $[t1, t2]$ and the time interval of event D be $[t3, t4]$. The meaning of addition of two time intervals is the time elapsed after these two events having occurred. The addition of two time intervals is defined as follow: $[t1, t2] + [t3, t4] = [t1 + t3, t2 + t4]$. Figure 6 shows the addition of two time intervals.

The time interval represents the remaining time of an occurrable event in the timed global state reachability analysis. Therefore, the purpose of the subtraction is to calculate the new remaining time of an occurrable event. When an event occurs, the remaining time of each of the other originally occurrable events that do not occur is defined as follows: Let the remaining time of the occurrable event C be $[t_1, t_2]$ and the remaining time of the occurrable event D be $[t_3, t_4]$. Let event

---

[1] However, if non-FIFO transmission is allowed, there is no non-FIFO error.

Figure 6: The addition of two time intervals

(a)

(b)

Figure 7: The subtraction of two time intervals, (a) two time intervals are not overlapped; (b) two time intervals are overlapped.

C occur, the new remaining time of event D is $[t3, t4]$ - $[t1, t2]$=$[max(0, t3 - t2), max(0, t4 - t1)]$, where $t3 - t2$ indicates the minimum time units left and $t4 - t1$ indicates the maximum time units left after event C having occurred. Since time cannot be negative value, the maximal operation is taken. Figure 7 illustrates the subtraction of two time intervals.

3.3 The Temporal Precedence Matrix

Each global state is associated with a temporal precedence matrix in order to maintain the inter-event time relationships of the occurrable events in a global state. As that described in Definition 2.7, the relationships are built when any two occurrable events co-exist at the first time in the global state reachability analysis. We adopt the similar definitions as that used in [25] for the TCSM model. In this subsection, we describe how to compute the entries in the temporal precedence matrix and

(a) (b) (c)

Figure 8: (a) Event-1 occurs before event-2, (b) event-2 occurs before event-1, (c) event-1 and event-2 are overlapped.

what the entries mean.

A temporal precedence matrix is indexed by the occurrable events of the associated global state. Each entry indicates time constraint between two occurrable events at the progress of the specified system. The relationship of two time intervals of two occurrable events may have the following possibilities: separate, meet, and overlap. Let the time interval of event E_1 be $[t1, t2]$ and the time interval of event E_2 be $[t1', t2']$. If event E_1 and event E_2 are separated, then $t1' > t2$. If event E_1 meets event E_2, then $t1' = t2$. If event E_1 are event E_2 are overlapped, then $t1' < t2$ and $t1 < t2'$.

If occurrable event E_1 meets occurrable event E_2, then E_1 is said to properly precede E_2. However, if one occurrable event does not properly precede another occurrable event, then we can make one of them properly precede the other by shifting the time interval of the first one forward or backward, and the value of shifting is called *adjustment distance*. In a temporal precedence matrix, Entry $(event - i, event - j)$ denotes the adjustment distance of occurrable event i such that its remaining time properly precedes that of occurrable event j. On the other hand, entry $(event - j, event - i)$ denotes the adjustment distance of event j such that its remaining time properly precedes that of event i. Let the value of entry $(event - E_1, event - E_2)$ be $t1' - t2$, and the value of entry $(event - E_2, event - E1)$ be $t1 - t2'$. If the value of an entry $(event - E_1, event - E_2)$ in the temporal precedence matrix is positive, it means that the two occurrable events are separated and E_2 needs to wait for at least the time units specified by that value after E_1 having occurred; otherwise, it means that (1) the two occurrable events are overlapped and E_2 may occur concurrently with E_1, or (2) E_2 may occur before E_1. Figure 8-(a) shows the value of an entry $(event - E_1, event - E_2)$ to be positive. In this case, E_1 and E_2 are separated, and E_1 must occur before E_2. Figure 8-(b) shows E_2 must occur before E_1, because the entry $(event - E_1, event - E_2)$ is negative, and the entry $(event - E_2, event - E_1)$ is positive. Figure 8-(c) shows two overlapped events, in which both the entry $(event - E_1, event - E_2)$ and the entry $(event - E_2, event - E_1)$ are negative. For two overlapped occurrable events E_1 and E_2, E_1 can occur before or after E_2. In fact, if $t1' < t2$ and $t1 < t2'$, then the value of the entry $(event - E_1, event - E_2)$ is the minimum time (delay) that E_2 has to wait for occurring, and the negative value of entry $(event - E_2, event - E_1)$ is the maximum time (deadline) that E_2 can wait for occurring after E_1 having occurred.

3.4 Succeedingly Reachable Global States from a Global State

In order to generate next global states of a given global state, the occurrable events which can occur in the given global state should be identified. These occurrable events are identified as follows:

1. Among those occurrable events whose predicates are true, find the one that has the smallest upper bound of the remaining time in a global state. This event is called *mature event*. If there are more than one occurrable event whose predicates are true, and whose upper bounds are same and are the smallest, select one arbitrarily.
2. Find other occurrable events whose remaining time is overlapped with that of the mature event. These occurrable events may be able to occur concurrently with the mature event in the global state.
3. After the overlapped occurrable events are detected, the associated predicates are checked. The mature event plus all of the overlapped events whose predicates are true are called *succeedingly occurrable events*. Succeedingly occurrable events are executed to derive the next reachable global states from a given global state. However, if the occurred event is not the mature event, i.e., an overlapped occurrable event with the mature event, then the maximum time that event can occur must be adjusted to that of the mature event due to the upper bound of the remaining time of the mature event is the smallest. That is, if an overlapped occurrable event occurs before the mature event, then the maximum delay of the overlapped event can not be beyond the upper bound of the mature event. Assume that three occurrable events, A, B, and C with remaining times of [3, 8], [4, 6], and [5, 9] are overlapped. Event B has the smallest upper bound so that it is the mature event. If event A occurs before event B, then the occurrence time of event A is chopped to [3, 6]. The remaining time of event B is then [0, 3], i.e., [4,6]-[3,6], after event A's occurrence. If event C occurs before event$B$, then the occurrence time of event C is chopped to [5, 6]. The remaining time of event B is then [0, 1], i.e., [4, 6]-[5, 6].

After the occurrence of an occurrable event, the state of the entity changes to the next state, the occurred event and those originally occurrable events which are not able to occur due to the occurred event are removed from the temporal precedence matrix, and the new occurrable event(s) of the new state are added into the generated global state matrix and its associated temporal precedence matrix. Additionally, the remaining time of the occurrable events is re-calculated in the generated global states. The removal of the associated events is examplified as follows: Let there be three outgoing transitions in state S_I, T_1 is a when transition, and T_2 and T_3 are spontaneous transitions. Assume that a message has arrived and T_1 can receive the message, then T_1 occurs and the state of the entity changes from S_I to S_I'. The corresponding events T_2 and T_3 are therefore removed from the temporal precedence matrix of the newly generated global state containing S'_I.

In our scheme, if event E_1 meets event E_2, we consider that event E_1 occurs before event E_2 and event E_2 is not overlapped with event E_1.

3.5 Timed Global State Reachability Algorithm

In this subsection, the timed global state reachability analysis algorithm for the TCSM model is presented. Before presenting the algorithm, the derivation of each occurrance event's remaining time and the derivation of the temporal precedence matrix of the new global state are described.

When an occurrable event occurs, the remaining time of the other occurrable events needs to be modified. Let occurrable event P's remaining time be $[p, p']$, and occurrable event Q's remaining time be $[q, q']$. Assume P represents the occurred event and Q is not the occurred event. In the temporal precedence matrix, t_{PQ}, i.e., entry $(event - P, event - Q)$, denotes the minimum time, and $-t_{QP}$, i.e., entry $(event - Q, event - P)$, denotes the maximum time that Q can still remain not occurring after P's occurrance. Due to the occurrence of P, an interval of time $[p, p']$ must have elapsed. Thus, the new remaining time of occurrable event Q is $[t, t']$=$[q, q']$-$[p, p']$ after P's occurrence. As a result, occurrable event Q must satisfy both $[t, t']$ and $[t_{PQ}$, $max(0, -t_{QP})]$ restrictions. Therefore, the correct new remaining time of occurable event Q in the new global state is $[max(t, t_{PQ}), min(t', max(0, -t_{QP}))]$.

Assume a child global state CG is generated from the parent global state PG. Let PT be the temporal precedence matrix of PG and CT be the temporal precedence matrix of CG. The derivation of the temporal precedence matrix CT of the newly generated global state GS is described as follows.

Procedure TPMNS: (Compute the Temporal Precedence Matrix of the Next Global State)

```
for each global transition E such that $PG \xrightarrow{E} CG$ do
    CT:=PT
    delete the entries that are associated with E from CT;
    if there are some originallly occurrable events that are prevented from occurring due to the
        occurrence of E
    then
        for each such E' do
            delete the entries that are associated with E' from CT;
        endfor
    endif
    insert the new occurrable events that result from the occurrence of E into CT;
    compute the entries that result from the insertion of new occurrable events in CT;
enddo;
```

An example of the temporal precedence matrix generation is shown in Figure 9. Let P, Q, and R be three occurrable events in a global state GS_X and their remaining time are [7,8], [40,40], and [8,10] respectively in GS_X. It is obvious that P can occur first. Let $GS_x \xrightarrow{P} GS_y$. Assume that after P having occurred, W is a new occurrable event in the newly generated global state GS_Y, and W's (initial) remaining time is [7,12]. According to the TPMNS procedure mentioned above, each timing constraint between two occurrable events is computed, and each entry in the temporal precedence matrix of GS$_y$ precisely expresses the adjustment distance of two occurrable events such that the row event can properly precede the column event.

Figure 9: An example of the computation of a temporal precedence matrix.

The initial global state is defined as below:

- Global state matrix : All channel entities are empty, each communicating entity is in its initial state, and all of the context variables of each communicating entity are in their initial values respectively.
- Temporal precedence matrix: All entries are filled with the corresponding adjustment distances of the initially occurrable events.

Let UGS be the global state pool storing unexplored global states, EGS be the global state pool storing explored global states, erroneous global states, and duplicated global states. The algorithm of timed global state reachability analysis is depicted in Appendix A.

At the beginning of the Timed Global State Reachability Analysis (TGSRA) that is in Appendix A, UGS and EGS are set to be empty and the initial global state is added to the unexplored global state pool UGS. Then, each global state GS in UGS is checked one by one in the **while** loop. In the while loop, an unexplored global state GS is removed from UGS and added to explored global state pool EGS at first. Next, all of the succeedingly occurrable events of GS are explored. In "Step a", if there is no succeedingly occurrable event, it means that GS contains a deadlock error. If GS is not a deadlock global state, then check whether GS contains a transmitted lock error or not. If GS is error-free, then "Step b" is executed.

In "Step b", all succeedingly occurrable events of GS are executed. The corresponding global transitions can be type S, type C, or type T. If the global transition belongs to type S, i.e., containing a spontaneous transition, then the peered channel entities receive the messages that are sent from communicating entities, and new global state(s) GS'(s) is (are) generated according to the associated channel transitions, e.g., some messages are transmitted successfully, some messages are lost or garbled, etc. Each newly generated global state GS' is checked whether there is a channel overflow error, or a non-FIF error if the communication is restricted to be First-In-First-Out (FIFO). If GS' does not contain channel overflow or non-FIFO error and GS' is not a duplicated global state, then GS' is added to UGS for exploration; otherwise, GS' is added to EGS.

If the global transition belongs to type C, i.e., containing a channel transition, then the peered communicating entity that should receive the message is checked whether there is an unspecified reception error. If the generated global state GS' contains an unspecified reception error, then add GS' to EGS; otherwise, the execution can be classified into two types according to the corresponding "when" transitions of the peered communicating entity: type C-I and type C-II. In the first case, a new global state GS' is generated and added to UGS if GS' is not a duplicated global state; otherwise, GS' is added to EGS. In the second case, all messages in the output events are sent to the corresponding channel entities. According to the associated channel transitions that transform the messages, each newly generated global state GS' is checked: if GS' has a channel overflow error or a non-FIFO error, then add GS' to EGS; otherwise, add GS' to UGS.

If the global transition belongs to type T, i.e., containing a timer transition, then a new global state GS' is generated. If GS' does not contain the premature timeout error and GS' is not a duplicated global state, then add GS' to UGS; otherwise, GS' is added to EGS.

The conditions for deadlock, unspecified reception, channel overflow, and transmitted lock errors, are described in Section 3.1. The procedure for detecting premature timeout is depicted in Appendix B.

In the procedure of "Detect a Premature Timeout (DPT)" that is in Appendix B, a premature timeout is validated backwardly starting from the global transition containing a timer transition in "Step a". If a global transition contains a response message loss channel transition, in which the response message is for responding the message sent by the communicating entity at which the timer transition is generated, then the timer transition is valid. The **while** loop in "Step a" is executed until the global state GS generating the timer transition is reached. In "Step b", if GS's incoming global transition contins a message loss channel transition in which the timer transition is associated with, then the timer transition is valid; otherwise there is a premature timeout error. An example of the premature timeout error is depicted in Figure 11, in which global state GS 20 contains a premature timuout error.

3.6 Examples

Figure 10 shows a modified Initiator-Responder protocol.[3] There are four communicating entities, i.e., Source, Initiator, Sink, and Responder, and there are six channel entities, i.e., ChanStoI (the Source to Initiator channel), ChanItoS (the Initiator to Source channel), ChanStoR (the Sink to Responder channel), ChanRtoS (the Responder to Sink channel), ChanItoR (the Initiator to Responder channel), and ChanRtoI (the Responder to Initiator channel). The channels between (1) Sink and Initiator, and between (2) Source and Responder, are reliable. The channels between Initiator and Responder are unreliable such that messages may be lost or garbled. Figure 11 shows the global state matrices part of the timed global state reachability analysis of the initiator-responder protocol. Figure 12 shows the temporal precedence matrices part of the timed global state reachability analysis of the initiator-responder protocol. In the Initiator-Responder protocol, there are some premature timeout and unspecified reception errors. The premature timeout error results from the unprecise time bound specified in transition T_1 of the Initiator entity. The cause of unspecified reception error is as follows: the Responder entity still may receive a PDU_CR at state Connected even after the connected confirm message, i.e., PDU_CC, having been sent. Thus, a self-loop transition, i.e., ?ChanItoR.PDU_CR/!ChanRtoI.PDU_CC, from state Connected to state Connected is required in the Responder entity.

4 The Estelle-based Timed Protocol Verification System (ETPVS)

Using the TCSM model and the associated timed verification scheme, an Estelle-based Timed Protocol Verification System (ETPVS) has been developed on SUN SPARC workstations. In this section, the Estelle Formal Description Technique (FDT) and the subset of Estelle that can be accepted by ETPVS are briefly introduced at first. Next, the main components of ETPVS are presented, Then, the functionalities provided in ETPVS are described.

4.1 Overview of Estelle

Estelle [7] is an FDT that is based on the extended state transition model, e.g., the ECFSM model. Using the module structures, production-rule-like control structures, and Pascal-based statements, Estelle is able to formally specify communication protocols. A specification in Estelle consists of a set of modules that can communicate with each other. Each module is specified as an ECFSM using facilities in Estelle. Each entity is described as a module. Communicating entities communicate with each other by exchanging messages through channels. Each channel is a FIFO queue that transmits interactions between two connected modules. Each transition consists of the following clauses: the FROM clause represents the head state, the TO clause represents the tail state, the PROVIDED clause represents a predicate which must be true so that the transition can be selected and executed, the WHEN clause represents an input interaction, the DELAY clause represents timing constraint in a transition, and the action part that is delimited by BEGIN and END key words. Figure 13 shows the abstract format of an Estelle-based specification.

The major aim of the TCSM model is providing a timed protocol specification model that can be directly applied to ISO's Estelle. In this way, protocols with time properties can be specified and automatically analyzed using Estelle. For precisely describing the behaviors of the channel entities, we have minor change of Estelle. Originally, a "when" transition is not associated with the delay clause. But a "when" transition is allowed to have the delay clause in our application. In this way, the channel transition can be specified in Estelle. Based on the modification, a channel entity can be represented as a module in Estelle. Consequently, a protocol specified in the TCSM model can also be mapped to a corresponding Estelle specification.

In order to have fully automatic execution, ETPVS accepts a subset of Estelle. Some restriction are as follows:

- The *systemprocess* and *process* modules are removed because global synchronization for *systemprocess* modules may lead to undesired overspecifications [8, 9]. Only *systemactivity* and

---

[3] The original Initiator-Responder protocol is in [16].

Figure 10: The modified Initiator-Responder protocol.

Figure 11: The global state matrices part of the timed global state reachability analysis.

Figure 12: The temporal precedence matrices part of the timed global state reachability analysis.

Figure 13: The abstract format of an Estelle-based specification.

activity modules are supported to have the execution of interleaving semantics [1].

- Dynamic features are not allowed, i.e., module instances and connections cannot be created dynamically. In other words, a static configuration is supported.
- In order to have fully automatic execution, incomplete definitions of functions/procedures, "...", and "any" variable types are not allowed.

4.2 Main Components of ETPVS

Figure 14 shows the abstract functional flow architecture of the ETPVS. There are two main components in ETPVS: an Estelle translator and a timed global state reachability analyzer. The Estelle translator has the following components:

1. A TCSM table generator: The TCSM table generator generates a link list that records the module specifications in an Estelle specification.
2. A module body generator: The body definition records the states, transitions, and the conditions that transitions can occur. The generator translates the body definitions into the internal structures for analysis.

Figure 14: The abstract functional flow architecture of ETPVS.

3. An interpreter: The interpreter interprets the condition part and the action part of each transition.

The global state analyzer executes all of the possible transition sequences according to the timed global state reachability analysis presented in Section 3.5. There are several components:

1. A global state initializer: The global state initializer initializes global state and temporal precedence matrices in order to generate the initial global state.
2. A queue initializer: The queue initializer constructs a queue for any two connected entities. From the connect statement in the initialization part of an Estelle specification, the links between different interaction points of different entities can be derived. According to the information, the connected queues can be constructed.
3. An occurrable event generator: The occurrable event generator decides the new occurrable events that should be put into the occurrable event list.
4. A succeedingly occurrable event selector: The succeedingly occurrable event selector selects all succeedingly occurrable events. When an unexplored global state is to be explored, all of the possible succeedingly occurrable events are calculated at first, then each succeedingly occurrable event is executed to generate new global states. The selector selects the succeedingly occurrable events according to the selection method described in Section 3.4.

5. A remaining time calculator: The remaining time calculator computes the remaining time of the occurrable events. When an occurrable event occurred, the remaining time of all of the other occurrable events is updated according to the computation method described in Section 3.5.
6. A global state matrix generator: The global state matrix generator generates the corresponding global state matrix for each newly generated global state.
7. A temporal precedence matrix generator: The temporal precedence matrix generator generates the corresponding temporal precedence matrix for each newly generated global state.
8. An error-check processor: The error-check processor checks the existence of errors and the error type in a global state.
9. A duplicated global state checker: The duplicated global state checker checks whether a newly generated global state is a duplication of the existed ones or not.

4.3 Functionalities of ETPVS

ETPVS provides a Graphic User Interface (GUI) based on the OPENLOOK X window system, The provided functions are as follows:

- To load the files containing Estelle specifications for analysis.
- To check the syntax correctness of Estelle specifications and to translate the Estelle specifications into the internal data structures.
- To set up the execution configuration, e.g., (1) the channel bound, (2) a halt point, i.e., the number of explored global states or the number of erroneous global states which can temporarily halt the execution of the timed global state reachability analysis. Figure 15 shows an example of setting up the execution configuration.
- To explore reachable global states according to the execution configuration. ETPVS interactively displays the status of up-to-date execution, e.g., the number of currently existed erroneous global states. Figure 16 shows an example of the execution status of the timed global state reachability analysis.
- To display the analysis result, e.g., displaying all erroneous global states and their error types, or displaying those global states containing some specific kinds of errors. Figure 17 shows an example of displaying analysis result.
- The help function provides the on-line help.

5 Discussion and Conclusion

Our TCSM model is in fact inspired by Lin and Liu's timed verification model [25]. In Lin and Liu's model, the time is associated with the head state of each transition. Under this definition,

Figure 15: Setting up the execution configuration.

Figure 16: Current execution status.

Figure 17: Displaying analysis result.

each state can have either (1) only one outgoing spontaneous transition for deterministic selection, or (2) more than one outgoing spontaneous transition with nondeterministic selection. As a matter of fact, the time cannot be associated with the transition in Lin and Liu's model. The main reason is that the transition with higher time bound is never able to be selected if the time of different transitions are not overlapped. If two or more outgoing transitions with different time intervals are specified in a state using Lin and Liu's model, since there is no predicate specification, the outgoing transition that has the lower upper bound is always fired first, and the others have no chance to be fired. In the TCSM model, the time interval is not associated with the state but with the transition. The reason that the time interval can be associated with the transition in the TCSM model is the existence of predicates. In the TCSM model, the predicate can prevent some transitions that have the lower bound of time intervals from being fired and can result in the execution of some transitions that have the higher bound of time intervals. Let T_1 and T_2 be the outgoing transitions of S_I. T_1 is associated with the time bound [1, 4] and a predicate $x \leq 2$, and T_2 is associated with the time bound [5, 6] and a predicate $x > 2$. If the predicate is not considered, T_1 is always selected as the mature event because its upper bound is the smallest. As a result, T_2 will not be selected forever. However, when the predicate is considered, if x is greater than two, T_2 can be selected as the mature event.

Global state reachability analysis always suffers from the state explosion problem. Some approaches have been proposed to relieve the state explosion problem. One approach is using some heuristics to reduce the explosion state space, e.g., state reduction techniques, or speedup the state explosion. Some non-timed EFSM-based state reduction techniques, which are suitable for Estelle-specified protocols, are proposed in [18, 22, 24]. An incremental verification technique [17, 19] and a database-oriented technique [12, 13] also have been adopted to speedup the state explosion. The other approach is using the partial verification approach, i.e., random walk [35] and the probabilistic verification technique [11, 20, 21, 28]. To relieve the state explosion problem in the TCSM model, we have proposed a probabilistic technique to have partial timed protocol verification [23].

We have adopted Estelle's "DELAY" clause into the TCSM model. From the other viewpoint, the "DELAY" clause in Estelle's language construct allows the application of the TCSM model and the associated timed global state reachability analysis. As a result, the applicable domain of Estelle is enlarged to formally specify and formally verify timed properties in protocols. Based on the TCSM model, we have developed an Estelle-based Timed Protocol Verification System (ETPVS) on SUN SPARC workstations. Timed protocols can therefore be specified in Estelle and be verified automatically. The future work falls into two aspects: (1) Based on the idea of adding animation to Estelle [2, 29], some graphics and animation mechanisms can be incorporated into the TCSM model. In this way, visualized timed protocol verification tools can be achieved. (2) Based on the TCSM probability-based partial timed protocol verification technique, extend the TCSM model to have performance analysis for timed protocols.

References

[1] B. Algayers, V. Coelho, L. Doldi, H. Garavel, Y. Lejeune and C. Rodriguez, "VESAR: a Pragmatic Approach to Formal Specification and Verification", *Computer Networks and ISDN Systems 25*, pp. 779-790, 1993.

[2] P. D. Amer and D. New, "Protocol Visualization in Estell," *Computer Networks and ISDN Systems 25*, pp. 741-760, 1993.

[3] F. Belina and D. Hogrefe, "The CCITT-Specification and Description Language SDL," *Computer Networks and ISDN Systems 16*, pp. 311-341, 1988.

[4] T. Bolognesi and E. Brinksma, "Introduction to the ISO Specification Language LOTOS," *Computer Networks and ISDN Systems 14*, pp. 25-59, 1987.

[5] T. Bolognesi and H, Rudin, "On the Analysis of Time-Dependent Protocols by Network Flow Algorithms,"*Proc. of Protocol Specification, Testing, and Verification, IV*, pp. 491-513, 1985.

[6] D. Brand and P. Zafiropulo. "On Communicating Finite-State Machines," *Journal of ACM*, Vol. 30, No. 2, pp. 323-342,1983.

[7] S. Budkowski and P. Dembinski, "An Introduction to Estelle: A Specification Language for Distributed Systems," *Computer Networks and ISDN Systems 14*, pp. 3-23, 1987.

[8] J. P Courtiat, "Estelle*: A Powerful Dialect of Estelle for OSI Protocol Description", *Proc. of Protocol Specification, Testing and Verification VIII*, pp. 171-186, 1988.

[9] J. P. Courtiat and Pierre de Daqui-Sannes, "ESTIM: an Integrated Environment for the Simulation and Verification of OSI Protocols Specified in Estelle*", *Computer Networks and ISDN Systems 25*, pp. 83-98, 1993.

[10] M. Diaz, "Modeling and Analysis of Communication and Cooperation Protocols Using Petri Net Based Models," *Computer Networks*, Vol. 6, pp. 419-441, 1982.

[11] D. D. Dimitrijevic and M. S. Chen, "A Procedure for Probabilistic Protocol Verification and Evaluation," *IEEE Transactions on Communications*, Vol. 40, No. 7, pp. 1183-1191, 1992.

[12] O. Frieder and G. E. Herman, "Protocol Verification Using Database Techniques," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 3, pp. 324-334, 1989.

[13] O. Frieder, "A Parallel Database-Driven Protocol Verification System Prototype," *Software Practice and Experience*, Vol. 22, No. 3, pp. 245-264, 1989.

[14] A. Gabrielian and M. Franklin, "Multilevel Specification of Real-Time Systems," *Communications of ACM 34*, pp. 51-60, 1991.

[15] G. Ghezzi, D. Mandrioli, S. Morasea, and M. Pezze, "A Unified High-Level Petri Net Formalism for Time-Critical Systems," *IEEE Transactions on Software Engineering*, Vol. 17, No. 2, pp. 160-172, 1987.

[16] D. Hogrefe, "OSI Formal Specification Case Study: The INRES Protocol and Service", Technique Report IAM-91-012, University of Berne, Institute of Computer Science and Applied Mathematics, 1991.

[17] Chung-Ming Huang and Jeng-Muh Hsu, "An Incremental Protocol Verification Method," *The Computer Journal*, Vol. 37, No. 8, pp. 698-710, 1994.

[18] Chung-Ming Huang and Jeng-Muh Hsu, "EHPVS: A Protocol Verification System for Verifying Protocols Specified in Estelle," *accepted by Journal of Chinese Institute of Engineers*, 1994.

[19] Chung-Ming Huang, Jeng-Muh Hsu, Huei-Yang Lai, Duen-Tay Huang, and Jao-Chiang Pong, "An Estelle-based Incremental Protocol Design System," *accepted by Journal of Systems and Software*, 1994.

[20] Chung-Ming Huang, Shiun-Wei Lee, and Jeng-Muh Hsu "Probabilistic Fuzzy Timed Protocol Verification," *accepted by Computer Communications*, 1994.

[21] Chung-Ming Huang, Shiun-Wei Lee, and Jeng-Muh Hsu, "ECFSM-based Probabilistic Protocol Verification," *accepted by Information Processing Letters*, 1995.

[22] Chung-Ming Huang, Jeng-Muh Hsu, Huei-Yang Lai, and Duen-Tay Huang, "An Integrated FDT-based Protocol Verification System," *accepted by IEE Software Engineering Journal*, 1995.

[23] Chung-Ming Huang, Shiun-Wei Lee, and Jeng-Muh Hsu, "Probability-based Partial Timed Protocol Verification" *submitted for publication.*

[24] F. J. Lin, P. M. Chu, and M. T. Liu, "Protocol Verification Using Reachability Analysis: the State Space Explosion Problem and Relief Stratefies," *Proc. of ACM SIGCOMM Workshop,* pp. 126-135, 1987.

[25] F. J. Lin and M. T. Liu, "An Integrated Approach to Verification and Performance Analysis of Communication Protocols," *Proc. of Protocol Specification, Testing, and Verification, VIII*, pp. 125-140, 1988.

[26] Y. J. Lin and G. Wuu, "A Constrained Approach for Temporal Intervals in the Analysis of Timed Transitions," *Proc. of Protocol Specification, Testing, and Verification, XI*, pp. 215-230, 1991.

[27] M. T. Liu, "Protocol Engineering," *Advances in Computers*, Vol. 29, Academic Press Inc., pp. 79-195, 1989.

[28] N. F. Maxemchuk and K. K. Sabnani, "Probabilistic Verification of Communication Protocols," *Proc. of Protocol Specification, Testing, and Verification* VII, pp. 307-320, 1987.

[29] D. New and P. D. Amer, "Adding Graphics and Animation to Estell," *Information and Software Technology*, Vol. 32, No. 2, pp. 149-161, 1990.

[30] W. X. Peng and S. Puroshothaman, "Datea Flow Analysis of Communicating Finite State Machines," *ACM Trans. on Programming Languages and Systems*, Vol. 13, No. 3, pp. 399-422, 1991.

[31] H. Saito, T. Hasegana, and T. Kakudu, " Protocol Verification System for SDL Specification based on Acyclic Expansion Algorithm and Temporal Logic," *Proc. of the 4th International Conference on Formal Description Techniques (FORTE'91)"*, pp. 513-528, 1991.

[32] B. Sajlowski, "On Verifying Time-Dependent Protocols,"*Proc. of International Conference on Software Engineering for Telecommunication Switching Systems*, pp. 46-51, 1986.

[33] A. U. Shankar and S. S. Lam, "Specification and Verification of Time-Dependent Communication Protocols,"*Proc. of Protocol Specification, Testing, and Verification, IV*, pp. 215-227, 1985.

[34] A. C. Shaw, "Communicating Real-Time State Machines,"*IEEE Transactions on Software Engineering*, Vol. 18, No. 9, pp. 805-816, 1992.

[35] C. H. West, "Protocol Validation by Random State Exploration," *Proc. of Protocol Specification, Testing, and Verification VI,* pp. 233-242, 1987.

APPENDIX A

Algorithm TGSRA: (Timed Global State Reachability Analysis)

```
Set UGS and EGS be empty;
add the initial global state into UGS;
while UGS is not empty do
    remove an unexplored global state GS from UGS and add GS to EGS;
    find the succeedingly occurrable events of GS;
    Step a: if there is no succeedingly occurrable event
            then mark GS with a deadlock error;
            else
              check whether there is a transmitted lock error in GS;
              if it is true
              then mark GS with a transmitted lock error;
              endif
            endif /* end of Step a */
    Step b: if GS is error-free
            then for each global transition E such that $GS \xrightarrow{E} GS'$ do case E of
             (I) Type S:
                  check whether there is a channel overflow or a non-FIFO error in each GS'
                  if it is true
                  then mark GS' with a channel overflow error or a non-FIFO error and add GS'
                    to EGS;
                  else add GS' to UGS if GS' is not a duplicated global state; otherwise, add
                    GS' to EGS.
                  endif
             (II) Type C:
                  check whether E results in an unspecified reception error;
                  if it is true
                  then mark GS' with an unspecified reception error and add GS' to EGS;
                  else case E of
                     type C-I: add GS' to UGS if GS' is not a duplicated global state; otherwise,
                       add GS' to EGS;
                     type C-II: check whether there is a channel overflow error or a non-FIFO
                       error in each newly generated global state GS'; if it is true then mark GS'
                       with a channel overflow error or a non-FIFO error and add GS' to EGS;
                       else add GS' to UGS if GS' is not a duplicated global state; otherwise, add
                       GS' to EGS.
                     endcase
                  endif
             (III) Type T:
                  check whether E causes premature timeout;
                  if it is true;
                  then mark GS' with a premature timeout error and add GS' to EGS;
                  else add GS' to UGS if GS' is not a duplicated global state; otherwise, add
                    GS' to EGS;
                  endif
            endif /* end of Step b */
endwhile
```

APPENDIX B

Procedure DPT: Detect a Premature Timeout

Assume a global transition E that belongs to type C, i.e., E contains a timer transition, occurs in global state GS, i.e., $GS \overset{E}{\rightarrow} GS'$. Let GS_a be the parent global state of GS.

```
Step a:  while this timer transition is not generated at GS do
           if the global transition from $GS_a$ to GS contains a response message loss channel
              transition, in which the response message is for responding the message sent by
              the communicating entity at which the timer transition is generated
           then it is not a premature timeout error.
           else GS ← $GS_a$, and $GS_a$ is the parent global state of the new GS
           endif
         endwhile
Step b:  if the global transition from $GS_a$ to GS contains a message loss channel transition and the
         message is sent by the communicating entity at which the timer transition is generated
         then it is not a premature timeout
         else a premature timeout is identified.
         endif
```