7

A method to derive a single-EFSM from communicating multi-EFSM for data part testing

I. Hwang, T. Kim, M. Jang, J. Lee, S. Lee, H. Oh[*], *and M. Kim*[**]
Department of Electronic Engineering, Yonsei University
Seodaemun-Gu Shinchon-Dong 134, Seoul, Korea, 120-749
Phone: +82-2-361-2864, Fax: +82-2-312-4584
E-mail: {his, jyl}@nasla.yonsei.ac.kr
[*] *Electronics and Telecommunications Research Institute(ETRI)*
Yusong-Gu Kajong-Dong 161, Taijon, Korea, 305-350
E-mail: hsohs@pec.etri.re.kr
[**] *Korea Telecom Research Laboratories*
Socho-Gu Umyun-Dong 17, Seoul, Korea, 137-792
E-mail: mckim@sava.kotel.co.kr

Abstract

The necessity that test sequences are automatically generated from a protocol specification for the purpose of testing data flow in the implementation has been emphasized because the cost of the most existing data part testing strategies is prohibitively high. However, existing automatic test generation methods based on single-module structure are not applicable to real protocol having multi-module one. In this paper, we propose a method which transforms multi-module model into an equivalent single-module. Since the proposed method uses the reachability analysis technique, it can minimize semantic loss of the specification during the transformation process.

Keywords

Conformance testing, data part testing, test generation, single-module

Testing of Communicating Systems, M. Kim, S. Kang & K. Hong (Eds)
Published by Chapman & Hall © 1997 IFIP

1 INTRODUCTION

Correct implementation of the protocol and the interoperability of the heterogeneous system have been emphasized as computer communication is widely used. Protocol conformance testing is carried out to check the conformance of a protocol implementation under test(IUT) to the protocol specification that it implements. Conformance to a communication protocol or service is considered to be prerequisite for the correct interoperability of open system.

In conformance testing, tester cannot observe or control the insides of the IUT and testing is done by applying test cases to the IUT and observing the output from it. Thus, test coverage is determined by the test cases and they should test most part of the protocol. Recently, necessity of generating test cases automatically has been emphasized due to the complexity of the communication protocols and a large amount of automatic test case generation methods have been proposed(Lee and Lee 1991)(Chanson and Zhu 1993)(Li *et al.* 1994)(Chin *et al.* 1997).

From a given protocol specification written in formal description techniques(FDTs) such as Estelle(ISO 1989a), LOTOS(ISO 1989b), and SDL(ITU-T 1993), finite state machine(FSM) or extended FSM(EFSM) model is obtained. Then, test cases are generated based on this FSM model to test control flow of the protocol or on EFSM model to test both control and data flow of the protocol.

Early work on test case generation has been based on a single-module FSM or EFSM model. However, most protocols have multi-module structure and existing test case generation methods are not applicable to these protocols. Therefore, transformation of the multi-module structure into an equivalent single-module is required. In this paper, we propose a method to obtain a single-module from multi-module protocol using reachability analysis technique. Since the proposed method simulates the behaviour of the protocol, semantic loss of the protocol during the transformation process can be minimized.

Section 2 presents related work and the outline of this work. In section 3, preliminaries necessary for the proposed method and the transformation algorithm are presented. Section 4 contains empirical results and section 5 discusses test case generation methods from EFSM model. Finally section 6 concludes this paper.

2 RELATED WORK

Conformance testing is done by applying proper input to the IUT and observing the output from it. In other words, disregarding the internal structure of the implementation, testing follows black box test which is carried out just at the interface. It can be divided into two categories; control part testing and data part testing. Control part testing is to test the control flow of the protocol based on FSM model to examine that transitions of the FSM are implemented correctly; to check whether there exist transition errors which are errors in the output function and transfer errors which are the case when next state is not the one expected after transition. Data part testing is to test the data flow of the protocol based on EFSM model. Test sequences are generated referring to data flow graph, and methods to generate test sequences are divided into two categories; using functional program test technique(Sarikaya *et*

al. 1987) and using data flow analysis technique(Ural and Yang 1991)(Miller and Paul 1992)(Chanson and Zhu 1993)(Ramalingom *et al.* 1996).

In general, implementing environment, even though for the same protocol, may be different case by case. For this reason, most specifications do not fully describe functions of some procedures, which are implemented to be suitable to different environments. However, in the case that a function of a procedure is not described, this procedure could not be tested and data flow testing is also meaningless. Therefore, the minimum features to be implemented in a procedure should be described in the specification and tester must check the minimum items which are general to all environments. In this paper, protocol specification is assumed to be full specification where every part of the protocol is described.

During the testing process the entire protocol becomes an IUT, which is regarded as a black box. Since most protocols consist of multi-module, test sequences that are capable of testing multi-module should be generated. However, existing test sequence generation methods are for either single-FSM or single-EFSM so that generated test sequences are for single-module. Therefore, we must transform the given protocol model into an equivalent single-module to generate test sequences. A test architecture for a multi-module IUT is given in Figure 1(Linn 1990).

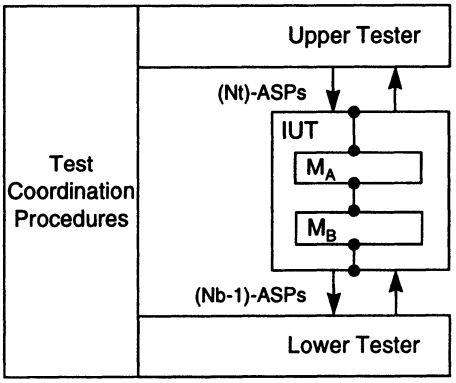

Figure 1 Test architecture for a multi-module IUT.

In Figure 1, messages given to or received from the IUT are called ASPs(Abstract Service Primitives). Tester cannot observe or control the interactions between M_A and M_B, and these interactions are called internal interactions. Interactions between M_A or M_B and the outside of the IUT are called external interactions which can be observed and controlled by the tester.

In (Sarikaya and Bochmann 1986), a single-module is obtained by textual replacement where internal communication is eliminated. For example, assume that an output interaction is transmitted to module M_B in a transition t_i of module M_A through internal channel invisible from the outside, and is received in a transition t_j of M_B. In this case, internal communication can be removed by substituting the output part of t_i with the action part of t_j. At the same time, the conditional statement of t_i should be properly modified according to the conditional statement in t_j.

In Estelle, firable condition of a transition is determined complicately. In the case that the protocol has hierarchical structure, by the parent/children priority principle, whether a transition in a child module can be fired depends on the firable transition in the parent module even though input interaction and conditional statement are satisfied(Dembinski and Budcowski 1989). However, with the method proposed in (Sarikaya and Bochmann 1986), it is difficult to consider this situation and there may occur some semantic loss during the transformation process.

In this paper, a method is proposed that simulates the behaviour of the protocol to obtain a single-module. This method adapts reachability analysis technique(West 1978) and a single-module structure is obtained as a result by simulating the behaviour of the protocol for all given external inputs. We assume that parameter values are considered when test sequences are generated. Therefore, we do not consider the parameter values of the external input during the transformation process and generated model includes all the possible cases according to the parameter values. Since the proposed method uses simulation, it can fully represent the behaviour of the protocol. It is assumed that non-determinism that makes the behaviour of the protocol complicated does not exist or can be eliminated by the tester(Lee and Lee 1991).

3 THE PROPOSED METHOD

3.1 Protocol Modelling

In Formal Methods in Conformance Testing(FMCT), a module of a protocol is modelled by an Input-Output State Machine(IOSM)(ISO 1995). IOSM is a 4-tuple $M =< S,\ L,\ T,\ s_0 >$ where S is a non empty finite set of states, L is a non empty finite set of interactions, $T \subseteq S \times (\{?,!\} \times L) \cup \{\tau\}) \times S$ is transition relation, and s_0 is the initial state of IOSM. Each element of transition relation T corresponds to a transition and has observable action such as input(?a) and output(!a), and internal action τ.

IOSM represents a protocol as an observable-based machine. However, it is not suitable to the automation of test case generation for data part testing based on formal model testing because the internal action τ is defined abstractly. Furthermore, it is difficult to identify which module is the destination when a module communicates with a lot of modules. In this paper, we introduce a communicating EFSM where the action block is simplified and interaction points are clearly defined. An interaction point is an interface of the communication.

Definition 1 Communicating ESFM model is a 9-tuple $M =< Q,\ q_0,\ I,\ O,\ V,\ P,\ A,\ \delta,\ IP >$ where

- Q is a non empty finite set of states;
- $q_0 \in Q$ is the initial state of M;
- I is a non empty finite set of input interactions;
- O is a non empty finite set of output interactions;
- V is a set of variables;

- P is a set of parameters;
- A is a set of actions;
- δ is a transition relation $\delta : Q \times A \longrightarrow Q$;
- IP is a non empty finite set of interaction points.

Each element of an action set A is a 4-tuple (*input*, *predicate*, *output*, *compute_block*). *input*, *output* are 3-tuple (ip_1, i, p_i), (ip_2, o, p_o), respectively, where ip_1, $ip_2 \in IP$, $i \in I$, $o \in O$, p_i, $p_o \in P$. *predicate* is a Pascal-like predicate expressed in terms of the variables and parameters. *compute_block* is a computation block which consists of linear functions $f : V \times P \longrightarrow V \times P$.

Note that input, output interactions and interaction points are defined in the action block of the CEFSM model. When a module communicates with a lot of modules, we can clearly identify which CEFSM is concerned with current communication due to the interaction points. Also note that the computation block is composed of linear equations to expedite the automation of test case generation and constraint solving. *compute_block* can be simplified using existing methods(Sarikaya and Bochmann 1986)(Miller and Paul 1992). In order to model hierarchical structure, CEFSM can be extended to 10-tuple by adding a property *ParentM* that represents the parent module of it.

A communicating system is a set of communicating EFSMs exchanging messages through FIFO(First In First Out) channels. In most cases, communicating system is made up of more than two communicating machines. In this paper, however, we will use only communicating systems composed of two communicating machines. It can be easily extended to general systems.

Definition 2 Communicating system is a 4-tuple $S =< CM_1,\ CM_2,\ C_{12},\ C_{21} >$ where

- $CM_i =< Q_i,\ q_{0i},\ I_i,\ O_i,\ V_i,\ P_i,\ A_i,\ \delta_i,\ IP_i >$, $i = 1, 2$ is CEFSM model;
- C_{ij}, $1 \leq i \neq j \leq 2$ is FIFO channel connecting interaction point ip_i and ip_j, where $ip_i \in IP_i$, $ip_j \in IP_j$.

The set M_{ij} is a set of messages from CM_i to CM_j and messages contained in a channel C_{ij} are represented as $c_{ij} \in (M_{ij})^*$. c_{ij} is the output message of CM_i and the input message of CM_j at the same time. As an example, when CM_i sends a message $c_{ij} = output_i = (ip_i,\ o_i,\ p_{oi})$ to CM_j, CM_j receives this messsage c_{ij} as an $input_j = (ip_j,\ i_j,\ p_{ij})$, where $i_j = o_i$, $p_{ij} = p_{oi}$.

The model obtained from communicating system through the transformation is called a global model. Global model is a directed graph $G = (V,\ E)$ where V is a set of global states and E corresponds to a set of global transitions. Global state and global transition are defined as follows.

Definition 3 A global state of a global model is a 2-tuple $g =< q_1,\ q_2 >$ where $q_i \in Q_i$ is the current state of CM_i.

Definition 4 A global transition of a global model is a pair $t = (i,\ \alpha)$ where $\alpha \in A_i$.

A global transition $t = (i, \alpha)$ is said to be firable in $g =< q_1, q_2 >$ if and only if the following two conditions are satisfied where $\alpha = (input, predicate, output, compute\_block)$ and c_{12}, c_{21} are messages contained in channel C_{12}, C_{21}, respectively.

- a transition relation $\delta_i(q_i, \alpha)$ is defined.
- $input = \epsilon$ and $predicate = True$ or
  $input = a$ and $c_{ji} = aw$, $w \in (M_{ji})^*$ and $predicate = True$.

After the global transition t is fired, the system goes to global state $g' =< q_1', q_2' >$ and messages contained in each channel are c_{12}', c_{21}' where

- $q_i' = \delta_i(q_i, \alpha)$, $q_j' = q_j$.
- if $input = \epsilon$ and $output = \epsilon$, then $c_{12}' = c_{12}$, $c_{21}' = c_{21}$
  if $input = \epsilon$ and $output = b = (ip_i, o_i, p_{oi})$, then $c_{ij}' = c_{ij}b$, $c_{ji}' = c_{ji}$
  if $input \neq \epsilon$ and $output = \epsilon$, then $c_{ij}' = c_{ij}$, $c_{ji}' = w$
  if $input \neq \epsilon$ and $output = b = (ip_i, o_i, p_{oi})$, then $c_{ij}' = c_{ij}b$, $c_{ji}' = w$.

A global model corresponds to an EFSM model. Therefore, we can get a single-EFSM from communicating multi-EFSM through the transformation of communicating system into global model.

3.2 Transformation Algorithm

Assume that a protocol is represented as in Figure 2. If module M_A and M_B are entities in (N)-layer, user A and user B are (N+1)-entities. Protocol specification written in formal method describes all the entities user A, user B, M_A, M_B, and transmission media. Each entity is a module or can be composed of multi-module as in Figure 2.

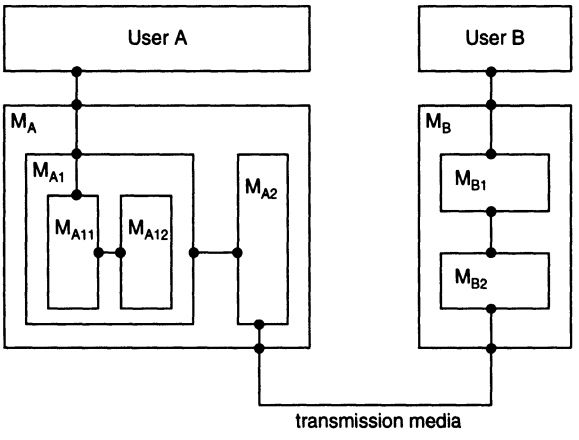

Figure 2 Protocol having hierarchical structure.

In order to test this protocol, IUT should be selected at first. In most cases, a protocol is designed to operate in at least one layer. Since the target of the testing is the implementation, minimum size of the IUT is the (N)-entity, M_A or M_B. Each of user A, user B, and transmission media can also be an IUT. An IUT may extend to more than one layer and M_A combined with transmission media can also be an IUT. In protocol specification, however, only definitions of interactions and simple functions to exchange interactions with other layers are defined for user A, user B, and transmission media. Therefore, we can assume that all the substantial functions of the protocol are in (N)-layer and it is sufficient to test only (N)-entity.

After selecting the module M_A as an IUT, we simulate the behaviour of the module M_A including all the child modules. During the simulation dynamic changing of the protocol structure may cause the size of the global model to get very large, so reduction of the size is needed. In this paper, we restrict the scope to the protocol that does not change its internal structure dynamically after initialization.

(Chun 1991) proposed a method that combines all modules one after another to generate a single-module. For example, the peer module M_{A11} and M_{A12} are combined using reachability analysis and single-module $M_{A11} + M_{A12}$ is obtained at first stage. In the second stage, parent module M_{A1} and the child module $M_{A11} + M_{A12}$ are combined, and then module $M_{A1} + M_{A11} + M_{A12}$ is generated. After repeating this procedure, all modules are combined and finally a single-module is obtained.

This method cannot fully reflect the behaviour of the protocol as in (Sarikaya and Bochmann 1986). As we have mentioned in section 2, the action of the child module can be affected by the parent module. However, if modules are combined one after another, it is difficult to consider this situation and the behaviour of the generated single-module may not be same to that of the protocol. In our method, actions of all the modules are simulated simultaneously so that the semantic loss during the transformation process can be minimized.

The obtained global model through the transformation should include all the behaviour of the communicating system and produce the same output for a given input. All the possible inputs are given to the protocol and reachability analysis technique is used to simulate the behaviour of the protocol. Simulation begins at the initial global state of the protocol*. For a given global state, we give all the possible external inputs to the protocol and check whether each transition is firable. Since we have assumed that the external input parameter values are determined during the test case generation, the value of the *predicate* that depends only on the external parameter is always *True*. For a firable transition, a directed edge from the current global state to the next global state is generated where the action block of the edge is same to that of the transition.

When all the external inputs are considered for the initial global state, the next global states moved from the initial state are taken into account. For each next global state, we check if there exist firable transitions. In this stage, we must consider the messages contained in the internal channel. Assume that there exists a message c_{ji} in the internal channel C_{ji} and a firable transition that receives the message c_{ji} as an input also exists. We can assume that the time required to process the internal interactions in a module is much shorter than the time interval between the external events because external events can be controlled by the tester. Then, there's no need

*Communicating system is initialized by the tester to consider the dynamic configuration of the protocol. More than one global states can be generated during the initialization phase.

to consider the external inputs and only transitions that receive no input or internal input are firable.

In the proposed method, messages in the channel are not contained in the global state and they only affect the firable condition of the transitions. This is the main difference with the reachability analysis technique where messages in the channel are containd in the global state. Essentially, the proposed method aims at obtaining a single-EFSM, not single-FSM. We just provide all the possible behaviour of the protocol and do not consider concrete values of the external parameters. Thus, state explosion that is the critical problem in verification technique where new states are generated for all input values does not occur, and the number of the generated global states is always finite. Transformation is completed when there is no global state to check. Algorithm for the transformation is as follows.

Algorithm

- Input: Communicating system S
- Output: Global model $G=(V, E)$

```
begin
initialize S
/* Communicating system S is initialized by the tester. Dynamic configuration
   of S is considered, and then some global states and edges are generated. */
for all generated global states g_k and edges e_k do
    begin
    V = V ∪ {g_k}
    E = E ∪ {e_k}
    end
V_CUR = V      /* global states to visit in current step */
V_NEXT = ∅     /* global states to visit in the next step */
while (V_CUR ≠ ∅) do
    begin
    for all global states g ∈ V_CUR do
        for all transitions t defined in g do
            begin
            if (firable transition t_i by null input exists)
            /* if at least one transition is firable without input, there's
                no need to consider any input. */
                for all transitions t_i
                    Process_This_Transition(t_i)
            else if (firable transition t_i by internal event exists)
            /* if at least one transition is firable by internal event,
                there's no need to consider any external input. */
                for all transitions t_i
                    Process_This_Transition(t_i)
            else
            /* all external inputs are considered. */
                for all transitions t_i firable by external event
                    Process_This_Transition(t_i)
```

```
            end
        V_CUR = V_NEXT
        V_NEXT = ∅
        end
    end

/* Process_This_Transition() decides whether this transition can be a new
   global edge. */
Process_This_Transition(t)
begin
    Create an edge e labelled by t.
    if (e ∉ E)
        begin
        E = E ∪ {e}
        if (next global state g' ∉ V)
            begin
            V = V ∪ {g'}      /* new global state */
            V_NEXT = V_NEXT ∪ {g'}
            end
        else
            V_NEXT = V_NEXT ∪ {g'}      /* existing global state */
        end
    else
        discard e      /* visited edge */
end
```

In (Chun 1991), global states having queue contents are divided into three categories; stable state, unstable state, and transient state.

- stable state: a state having no firable transition or no message in the internal queue
- unstable state: a state having firable transitions by internal events
- transient state: a state having firable transitions without any input

Tester cannot control the behaviour of the protocol in unstable and transient state. So, removing these states makes no difference to the behaviour of the protocol when we observe it from the outside. Since the global state in the proposed model does not include the queue contents, global states should be classified in different way.

- If there exists a firable transition without any input, this state is a transient state.
- If a state is not a transient state and there exists at least one firable transition that makes the system go to this state without giving any output to internal channel, this state is a stable state.
- If a state is not a transient state and there exists at least one firable transition by internal event, this state is an unstable state.

A global state may have properties of both stable and unstable state, and the removal of unstable state does not mean the removal of the global state. Unstable state can be removed by combining two transitions that communicate through internal channel, and this global state can be removed if it doesn't have the property of stable state. We can combine two transitions using symbolic execution technique(Clark and Richardson 1985).

4 EMPIRICAL RESULTS

In this paper, we have applied the proposed method to the Class 0 transport protocol(TP0). Modular structure of the TP0 is shown in Figure 3.

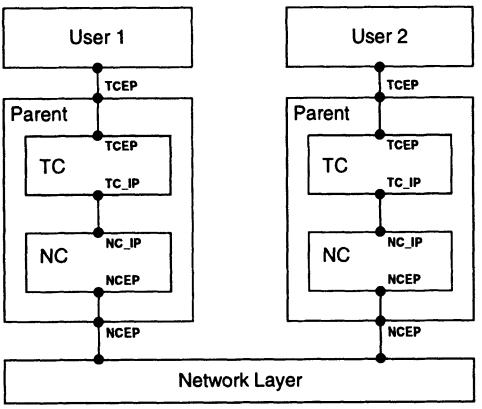

Figure 3 Modular structure of the TP0.

In Figure 3, module Parent initializes and releases the module TC, NC and internal channels, and delivers interactions that were sent from TC or NC to the external modules. TC makes TPDU(Transport Protocol Data Unit) for each input primitive and NC makes NSDU(Network Service Data Unit) for each TPDU to communicate with network layer.

As the target protocol of the testing is transport protocol, the IUT is module Parent, and three modules, Parent, TC, and NC should be combined to generate test sequences. In TP0, module Parent changes the dynamic configuration of the structure, so this should be excluded in the transformation process. However, Parent is in charge of establishment and release of the connection, so we will include this module partially to test the overall flow of the protocol. After the connection is established, we restrict the function of the module Parent to delivering interactions between module TC, NC and external modules. Then, TCEP of TC and NCEP of NC become the external interaction points where tester can observe or control the interactions

Transformation process is divided into two steps. First, we simulate the behaviour of the protocol and the global model is generated. At this time, communication is classified into internal one and external one. In the case of TP0, TCEP and NCEP are external interaction points and TC_IP, NC_IP are internal interaction points.

When TC receives disconnection request from the upper layer, TC makes a TPDU containing this information and sends it to NC. NC receives this TPDU and sends it to the lower layer after encoding it. In this process, communication between TC and NC is an internal communication that tester cannot observe or control. Figure 4 shows this procedure.

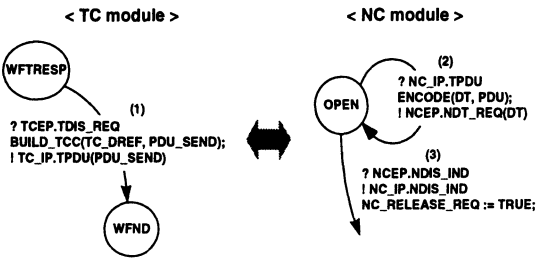

Figure 4 Internal communication in TP0.

Assume that TC, NC are in WFTRESP, OPEN state, respectively, and there's no message in the internal channel connecting TC_IP and NC_IP. The global state of this system is then, (WFTRESP, OPEN) and three transitions (1), (2), and (3) are related to this state. Since there's no message in the internal channel, transition (1), (3) are firable and global transitions labelled (1), (3) are generated. However, we will consider just transition (1) to simplify the explanation. At first, transition (1) is fired and global transition labelled (1) is generated. After the transition (1) is fired, the system goes to the global state (WFND, OPEN) and the message TPDU exists in the internal channel. Although two transitions, (2) and (3) are concerned with this state, only transition (2) is firable because (2) receives internal input while (3) receives external one. After the transition (2) is fired, the internal channel is emptied and the transition (3) becomes firable. Generated global model from Figure 4 is shown in Figure 5(a).

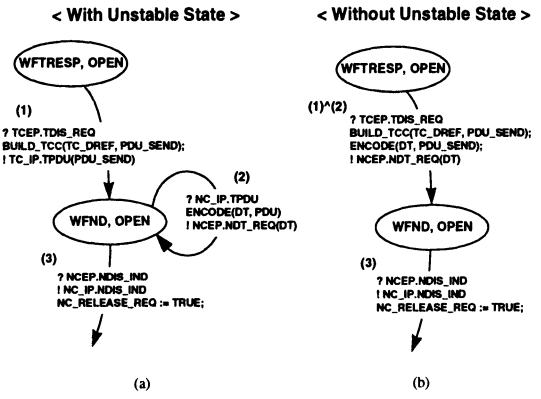

Figure 5 Global model with unstable state and without unstable state.

As a second step, transient states and unstable states are eliminated. In the (WFND, OPEN) state in Figure 5(a), transition (2) is firable by internal event and there's a transition that makes the system go to (WFND, OPEN) state without giving any internal event. Thus, (WFND, OPEN) state has the properties of both stable and unstable state. Unstable property of the state (WFND, OPEN) can be removed by combining two transitions, (1) and (2) which communicate through internal channel. Figure 5(b) shows the result after combining two transitions.

When there's no global state to consider, transformation ends and finally we can get the global model of the communicating system. Figure 6 shows the global model of TP0. In this figure, dotted line represents the dynamic changing of the protocol structure that should be processed manually. It is important to note that 4_1, 5_1, and 6_2 states are unremoved unstable states or transient states due to the specification where some part of the protocol are not described. The action blocks of the transitions are available in (Hwang 1997).

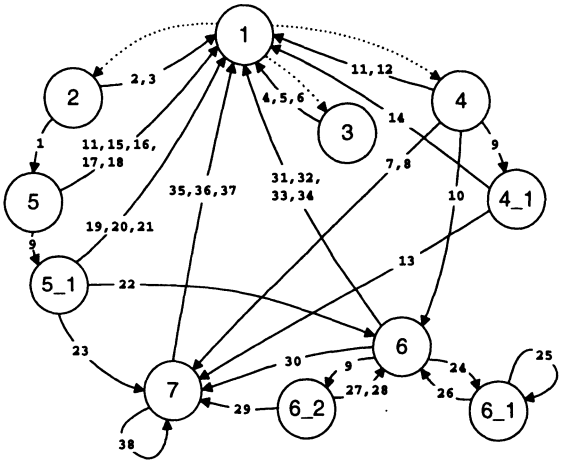

Figure 6 Global model of TP0.

5 TEST CASE GENERATION

Currently, there are two approaches to generate test sequences from EFSM. One is to generate test cases in respect to test purposes(Guerrouat and König 1996). A test case is derived from a test purpose which represents a control flow of the protocol. In this approach, only control flow is checked and data part of the protocol is used for generating executable test cases. Since test purposes are usually informal and may express different kinds of conformance requirements, it is impossible to automate this procedure. Therefore, formalization of the test purposes and automatic test case generation from the formally described test purposes are required. In (Guerrouat and König 1996), test cases are generated automatically for restricted classes of test purposes using a knowledge based techniques.

In the second approach, test sequences are generated automatically using algorithms. In order to test the control flow of the protocol, one can get FSM from the EFSM by exhaustive simulation with input parameter values. In most cases, however, it is not feasible to generate FSM for all input values, so compromise between the size of the generated model and the accuracy of the automation is required. To test the data flow of the protocol, some test sequence generation algorithms based on data part testing criteria are applied to EFSM(Chanson and Zhu 1993)(Li *et al.* 1994)(Ramalingom *et al.* 1996)(Chin *et al.* 1997). In general, the size of the generated test sequences is very large when they are generated automatically even though the target protocol is simple. As the size of the protocol increases, the size of the generated test sequences increases more rapidly. Thus, it is impractical to generate test sequences for complex protocol and the size of the target protocol should be reduced.

Protocol model can be divided into sub-graphs based on some criteria. (Park *et al.* 1994) proposed a method to reduce the size of the problem by applying test purposes to the protocol. Since a test purpose represents a control flow, it can be a sub-graph of the protocol and test sequences are generated based on this sub-graph. In order to automate this procedure, protocol should be divided systematically and manual effort should be minimized.

6 CONCLUSIONS

We have presented a method that transforms communicating multi-module protocol into an equivalent single-module to generate test cases for both control and data part testing. The transformation process is divided into two steps. First, we simulate the behaviour of the protocol for all the possible inputs, and then the global model is generated. As a second step, transient and unstable states are eliminated to reduce the size of the global model. Since the proposed method adapts reachability analysis technique rather than textual replacement, we can minimize the semantic loss of the protocol during the transformation process.

In order to apply the proposed method to the FDTs, a detailed study is needed for each FDT. Definition of the EFSM should be extended according to the FDT's properties and firable conditions should also be modified. Currently, we are extending our study to obtaining an EFSM from the specification written in a FDT and generating feasible test cases from the global model.

ACKNOWLEDGEMENTS

This work has been supported partially by the Electronics and Telecommunications Research Institute(ETRI) of Korea and partially by the Korea Telecom.

7 REFERENCES

Chanson, S.T. and Zhu, J. (1993) A Unified Approach to Protocol Test Sequence Generation. In *Proceedings of the IEEE INFOCOM*, San Francisco, Califor-

nia, 106–114.

Chin, B., Kim, T., Hwang, I., Jang, M., Lee, J., and Lee, S. (1997) Generation of Reliable and Optimized Test Cases for Data Flow Test with a Formal Approach. In *Proceedings of the 11th International Conference on Information Networking*, Taipai, Taiwan.

Chun, W. (1991) Test Case Generation for Protocols Specified in Estelle. *Ph.D. thesis*, Dept. Computer and Information Sciences, University of Delaware.

Clarke, L.A. and Richardson, D.J. (1985) Application of Symbolic Evaluation. *the Journal of Systems and Software*, **5**, 15–35.

Dembinski, P. and Budcowski, S. (1989) Specification Language ESTELLE in *The Formal Description Technique Estelle* (eds. M. Diaz *et al.*), Results of the ESPRIT/SEDOS Project, North Holland.

Guerrouat, A. and König, H. (1996) Automation of test case derivation in respect to test purposes. In *Proceedings of the 9th International Workshop on Testing of Communicating Systems*, Darmstadt, Germany, 207–222.

Hwang, I. (1997) A method to derive a simple single-EFSM from communicating multi-EFSM for data part testing. *M.S. thesis*, Dept. Electronic Engineering, Yonsei University.
URL: http://nasla.yonsei.ac.kr/~ his/Publications/Th_annex.ps

ISO (1989a), *Information Processing Systems - Open Systems Interconnection - Estelle: A Formal Description Technique based on an Extended State Transition Model.* ISO/IEC IS 9074.

ISO (1989b), *Information Processing Systems - Open Systems Interconnection - LOTOS: A Formal Description Technique based on the Temporal Ordering of Observational Behaviour.* ISO/IEC IS 8807.

ISO (1995), ISO/IEC JTC1/SC21 WG7, ITU-T SG10/Q.8 *Formal Methods in Conformance Testing Part, working draft.* ISO Project 1.21.54, ITU-T Z.500. ISO, ITU-T, September, 1995. Output from ISO/ITU-T Meeting in Geneve.

ITU-T (1993) ITU-T Z.100 *CCITT Specification and Description Language.* ITU Recommendation.

Kim, T. (1995) Automatic generation of observation-based and length-optimized test cases for EFSM model in conformance testing. *M.S. thesis*, Dept. Electronic Engineering, Yonsei University.

Lee, D.Y. and Lee, J.Y. (1991) A well defined Estelle specification for the automatic test generation. *IEEE Trans. on Computer*, **COM-40(4)**, 526–542.

Li, X., Higashino, T., Higuchi, M. and Taniguchi, K. (1994) Automatic generation of extended UIO sequences for communication protocols in an EFSM model. In *Proceedings of 7th International Workshop on Protocol Test Systems*, Tokyo, Japan, 225–240.

Linn Jr., R.J. (1990) Conformance Testing for OSI Protocols. *Computer Network and ISDN Systems*, **18**, 203–219.

Miller, R.E. and Paul, S. (1992) Generating Conformance Test Sequences for Combined Control and Data Flow of Communication Protocols. In *Proceedings of 12th International Symposium on Protocol Specification, Testing and Verification*, 1–15.

Park, J.H., Lee, J.Y., Jung, I.Y., and Hong, J.P. (1994) A Conformance Testing Framework for Applying Test Purposes. In *Proceedings of 7th International Workshop on Protocol Test Systems*, Tokyo, Japan, 291–298.

Ramalingom, T., Thulasiraman, K. and Das, A. (1996) Context Independent Unique Sequences Gerneration for Protocol Testing. In *Proceedings of the IEEE INFOCOM*, San Francisco, California, 1141–1148.

Sarikaya, B. and Bochmann, G.v. (1986) Obtaining Normal Form Specification for Protocols. *Computer Network Usage* (eds. L. Csaba *et al.*), 601–612.

Sarikaya, B., Bochmann, G.v. and Cerny, E. (1987) A Test Design Methodology for Protocol Testing. *IEEE Trans. on Software Engineering*, **SE-13**, 518–531.

Ural, H. and Yang, B. (1991) A Test Sequence Selection Method for Protocol Testing. *IEEE Trans. on Communication*, **COM-39**, 514–523.

West, C. H. (1978) An automated technique of communications Protocol Validation. *IEEE Trans. on Communication*, **COM-26**, 1271–1275.

8 BIOGRAPHY

Iksoon Hwang received the B.S. and M.S. degrees in electronic engineering in 1995 and 1997, respectively, from Yonsei University, Seoul, Korea. Currently, he is in the Ph.D. degree course in electronic engineering at Yonsei University. His current interests include protocol engineering, ATM networks, and computer networks.

Taehyong Kim received the B.S. and M.S. degrees in electronic engineering in 1993 and 1995, respectively, from Yonsei University, Seoul, Korea. He is currently a Ph.D. candidate in electronic engineering at Yonsei University. His current research interests include protocol engineering, software engineering, and computer networks.

Minseok Jang received the B.S. and M.S. degrees in 1989 and 1991, respectively, and Ph.D. degree in electronic engineering in 1997 from Yonsei University, Seoul, Korea. His current research interests include protocol engineering, software engineering, especially conformance testing, and computer networks.

Jaiyong Lee received the B.S. degree in electronic engineering in 1977 from Yonsei Universtiy, Seoul, Korea and M.S. and Ph.D. degrees in computer engineering in 1984, and 1987, respectively, from Iowa State University, Ames, Iowa. From 1977 to 1982, he was a research engineer at Agency for Depense Development of Korea. From 1987 to 1994 he was a Associate Professor at Pohang Institute of Science and Technology. He is currently a Professor in the Department of Electronic Engineering, Yonsei University, Seoul, Korea. He is interested in high speed/multimedia communication, multimedia PCS protocol, and conformance testing.

Sangbae Lee received the B.S. from the R.O.K. Air Force Academy of Korea in 1958. the B.E. degree in electrical engineering from the Seoul National University, Seoul, Korea, in 1961, the M.S. degree in from Standford University, California, in 1964, and the Ph.D. degree from the University of Newcastle, England in 1975. He is currently a Professor in the Department of Electronic Engineering, Yonsei University, Seoul, Korea. From 1969 to 1979, he was an Assistant Professor at Seoul National University and from 1982 to 1983, a Visiting Professor at the University of Newcastle, England. He has served as a Chairman of the IEEE Korea Section from 1986 to 1987 and as a Chairman of the Korea Institute of Telematics and Electronics. He is interested in B-ISDN, graph theory, LAN/MAN internetworking, and multimedia communication.

Heangsuk Oh received the B.S. and M.S. in eletronic material engineering from Hanyang University in 1981 and 1983, respectively, and Ph.D. in computer science from Chungbuk University in 1997. His research interests include computer network, formal description techniques, and protocol engineering. From 1993 to the present, he worked as a senior researcher of protocol engineering project where he serves in PEC(Protocol Engineering Center) at ETRI.

Myungchul Kim received B.A. in electronic engineering from Ajou Univ. in 1982, M.S. in computer science from the Korea Advanced Institute of Science and Technology in 1984, and Ph.D. in computer science from the Univ. of British Columbia in 1992. Currently he is with the Korea Telecom Research and Development Group as a managing director, Chairman of Profile Test Specification-Special Interest Group of Asia-Oceania Workshop, and is the Co-Chair of the 10th IWTCS'97. His research interests include protocol engineering on telecommunications and multimedia.