INFORMATICA, 1993, Vol.4, No.1-2, 199-216

SPECIFICATION, VALIDATION AND SIMULATION OF AN EVENT DRIVEN LOCAL COMPUTER NETWORK PROTOCOL

Henrikas PRANEVITCHIUS

Managment Informatics Department
Kaunas University of Technology
3028 Kaunas, Studentų St. 50, Lithuania

Leo SINTONEN

Software Systems Laboratory
Tampere University of Technology
Box 527, SF-33101 Tampere, Finland

Abstract. An analysis of an event driven local area network protocol for special purposes in indutrial applications is represented. The analysis is performed using the specification language ESTELLE/Ag and protocol analysis tool PRANAS-2. Validation was focused on the correctness media access algorithm, initialization of the protocol and recovery from error situations. The obtained simulation results can be used for comparison of the effectiveness of this protocol with other protocols.

Key words: specification, validation, simulation, local computer networks, high-speed protocols, specification language ESTELLE/Ag.

1. Introduction. There are many computer communication applications requiring high bandwidth and high reliability in operation, still making possible simple and low-cost implementation. This type of networks exists in robotics, vechiles, homes, etc. These applications set restrictions on system in terms of usable hardware, cost and cabling. Such networks are in many cases devoted to one special application and not to a general purpose use. The number of stations is typicaly small compared with typical LAN applications, and the variation of number of stations is small during the life cycle of the network.

Typical requirements for the media access protocols in these applications are: high reliability in the environment, where electrical disturbance level is high scalable bandwidth, self-stabilizing properties and simplicity combined with low cost of implementation. Solutions based on existing media access standards do not meet these requirements in many cases. The protocol described by Sintonen (1990) is designed to offer high bandwidth but keep the structure simple. The configuration is a physical bus, where station form a logical ring. The algorithm is based on noticeable events on the bus, hence the name "event driven bus protocol" Sintonen (1990). It belongs to a class of iplicit-token DAMA-schemes.

The protocol is distributed, except in the initialization phase. Every station listens to the bus and receives both the destination address and the source address, and stores them in the registers DA and SA respectively. A station also is capable of sending the bus and detecting the event "Frame ended". Algorithm for sending and receiving is as follows. Receiving: When a station notices it's own address in the DA field, it receives the frame. Sending: When a station has a frame to send, it waits until receives the address of it's predecessor in the SA of the frame. Then it waits for the event "Frame ended". After that event happened it waits a time period $D', D' <= 2 * d$, where d is the end to end delay of the bus. Then it sends it's frame. After the end of frame it wait for a time delay $D, D >= 2 * d$ to hear the next station begin sending. When this happens, the sending phase is ended. If a station has nothing to send when it's turn commes, it sends an empty "no data" frame, a kind of a token, to give the turn to the next station in sequence.

There is one station which initializes the ring. This is the fixed control station. The control station also detects failed station and is capable of executing an reconfiguration algorithm to restore the normal operation of the ring.

In this paper we represent the formal specification of this protocol based on the aggregative method (Pranevitchius, 1991). On the ground of this method protocols specification language ESTELLE/Ag and protocol analysis tool PRANAS-2 are created

(Pranevitchius, Pilkauskas and Chmieliauskas, 1991). There are some differences between ESTELE/Ag and the ESTELLE standard of ISO. Model of piece-linear aggregate is used (Pranevitchius, Pilkauskas and Chmieliauskas, 1991) in ESTELLE/Ag. The use of this model instead finite-state automaton, which is the formal background of the standard ESTELLE, gives us benefits we can express as the following main properties of the language:

1) the language is executable,
2) the language is multipurpose.

The first property requires leaving out some of the syntactical constructs, allowed in the ESTELLE standard, i.e. the the descriptive of the language is reduced, e.g. the dynamic change of the specification structure is not defined while solving validation (verification) and simulation problems.

Meanwhile the second property demands introduction of several new constructs, widening the application field of the language. This means that specifications written in this language can be used in solving such different protocol design problems as validation, verification and simulation (Pranevitchius, Pilkauskas and Chmieliauskas, 1991).

Each protocol module is considered to be a piece- linear aggregate. Therefore the inner functioning of the module is define in terms of a mathematical model of the aggregate. The current state of the module is determined by the following components:

1) an internal structure of the module,
2) a set discrete variables,
3) a set of continuous variables.

The newly introduced constructions are not related to the module strukture description. they are mainly related to the description of the module evoliution (i.e. components 2 and 3). The set of discrete variables is similar to the one used in most finite-state models, i.e. it consists of the so-called "basic" and "context" variables describing the changees in the module state due to the occurrence of events. Syntactically they are described in a variable declaration part of each body definition as the ISO's ESTELLE standard. The continous variables control the seguence of events. In ESTELLE/Ag these variables are called "operations".

Using the mentioned tools we created specification and performed validation of this protocol. In this validation we focus on the correctnes of the initialization of the operation, corrrectness of the normal operation sequence and correctness of the self-stabilation. The same specification was used for simulating of protocol. Our simulation gives a basis to estimate the benefits of the protocol.

The organization of this paper is as follows: Firstly, aggregate specification of protocols is represented, and after that some results of validation and simulation are represented.

2. Aggregative specification of protocol. An aggregative scheme of specification of analyzed event oriented protocol is depicted in Fig. 1. The agregates Station_0, Station_1,...,Station_(n-1) depict stations which are switched on to the network and aggregate Bus describes the performance channel. Station_0 is the controling one. The signals that are transmitted between the aggregates have been shown too. Formal description of the aggregates is executed according the approach represented by Pranevitchius (1991).

Aggregate station_nr, nr=1,n-1

1. Set of input signals

X_{nr} = {fr_end(m),bus_is_oc, no_data(m),fail},
where fr_end(m) – end transmitting of frame,
bus_is_oc – bus is ocuppied,
no_data(m) – no data for transmission,
st_on – switching on of the station,
n – number of stations.

2. Set of output signals.

Y_{nr} = {beg_fr,end_trans,no_date,fail},
where beg_fr – beginning of transmitting frame,
end_trans – end of transmitting frame,
no_data – no data for transmitting,
fail – station is switched off.

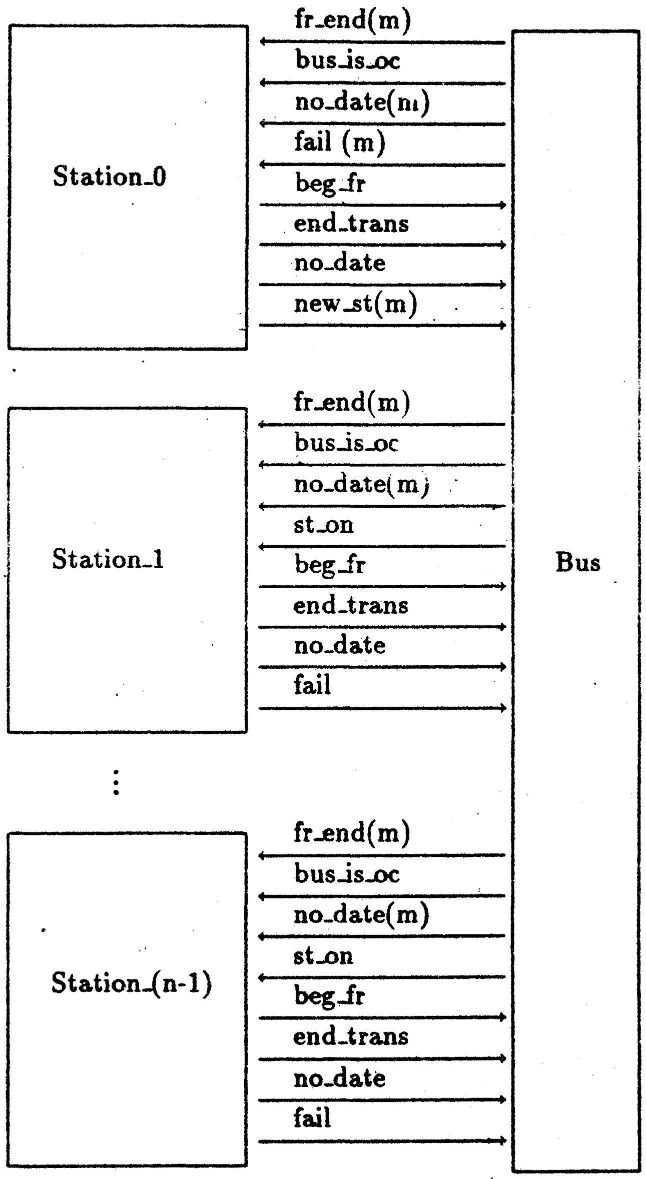

Fig. 1. Aggregate scheme of model.

3. Set of internal events.

$E''_{nr} = \{e''_1(\text{taim\_}DI), e''_2(\text{taim\_}D), e''_3(\text{trans\_}fr),$
$e''_4(\text{arr\_fr}), e''_5(\text{swit\_of}), e''_6(\text{no\_data})\}$,
where $e''_1(\text{taim\_}DI)$ – end of taimer DI;
$e''_2(\text{taim\_}D)$ – end of taimer D;
$e''_3(\text{trans\_fr})$ – end transmitting of frame;
$e''_4(\text{arr\_fr})$ – moment of arriving new frame;
$e''_5(\text{swit\_of})$ – moment of switching off station;
$e''_6(\text{no\_data})$ – no data for transmitting.

4. Controlling sequences:

$e''(\ldots) \to \{\xi_{ij}\}$, $i = \overline{1,6}$, $j = \overline{1,\infty}$, where ξ_{ij} – duration of operation, after that occurs event $e''(\ldots)$.

5. $\nu(t_m) = \{\text{st}(t_m), \text{act}D(t_m), \text{sw}(t_m)\}$,
where

$$\text{st}(t_m) = \begin{cases} 0, & \text{no frame for transmitting;} \\ 1, & \text{there is a frame for transmitting;} \end{cases}$$

$$\text{act}D(t_m) = \begin{cases} 0, & \text{time } D \text{ is switched off;} \\ 1, & \text{timer } D \text{ is switched on;} \end{cases}$$

$$\text{sw}(t_m) = \begin{cases} 0, & \text{station is switched off;} \\ 1, & \text{station is switched on.} \end{cases}$$

6. Initial state.

$\text{st}(t_0) := 0,\ \text{act\_}D(t_0) := 0,\ \text{sw}(t_0) := 0,$
$W(e''_5(\text{swit\_of})) := t_0 + \xi_{5j};$
$W(e''_4(\text{arr\_fr})) := t_0 + \xi_{4j};$
$W(e''_3(\text{trans\_fr})) := \infty;$
$W(e''_2(\text{taim\_}D)) := \infty;$
$W(e''_1(\text{taim\_}DI)) := \infty;$

7. Transfer operators:

```
$H[e'$ (fr_end)]:
    if sw($t_m$) = 1 and act_$D$($t_{m+1}$) = 1 then begin
      $W(e''$(taim_$D$), $t_m$) := $\infty$;
      act_$D$($t_m$) := 0;
      end;
    end;
$H[e'$(bus_is_oc)]:
    if sw($t_m$) = 1 and act_$D$($t_m$) = 1 then begin
      $W(e''$(taim_$D$), $t_{m+1}$) := $\infty$;
      act_$D$($t$) := 0;
      end;
    end;
$H[e'_3$(no_date)] :
    if sw($t_m$) = 1 then begin
      if $m = nr$ then $W(e''$(taim_$DI$), $t_{m+1}$) := $t_m + \xi_{1i}$;
      if act_$D$ = 1 then begin
        $W(e''_2$(taim_D), $t_{m+1}$) := $\infty$;
        act_$D$ = 0;
        end
      end
    end;
$H[e'_4$(st_on)] :
    st($t_{m+1}$) := 0;
    $W(e''_4(t_{m+1}), t_{m+1})$ := $t_m + \xi_{4i}$;
    $W(e''_5(t_{m+2}), t_{m+1})$ := $t_m + \xi_{5i}$;
    end;
$H[e''_1$(taim_$DI$] :
    $y_1$ := beg_fr;
    if st($t_m$) = 1 then
      $W(e''_3$(trans_fr), $t_{m+1}$)) := $t_m + \xi_{3i}$
    else
      $W(e''_6$(no_date), $t_{m+1}$)) := $t_m + \xi_{6i}$
    end;
```

```
$H[e_2''$ (taim_$D$)]:
        act_$D := 0$;
        end;
$H[e_3''$(trans_$Fr$)] :
        st$(t_{m+1}) := 0$;
        $W(e_2''$(taim_$D$)$, t_{m+1}) := t_m + \xi_{2i}$;
        act_$D := 1$;
        $y_2 :=$ end_trans;
        end;
$H[e_4''$(arr_fr)] :
        $W(e_4''$(arr_fr)$, t_{m+1}) := t_m + \xi_{4i}$;
        if st$(t_{m+1}) = 0$ then
              st$(t_m) := 1$
        end;
$H[e_5''$(swit_off)] :
       $y_4 :=$ fail;
       sw$(t_{m+1}) := 0$;
       $W(e_1''$(taim_$DI$)$, t_{m+1}) := \infty$;
       act_$D(t_{m+1}) := 0$;
       $W(e_2''$(taim_$DI$)$, t_{m+1}) := \infty$;
       $W(e_3''$(trans_fr)$, t_{m+1}) := \infty$;
       $W(e_4''$(arr_fr)$) := \infty$;
       st$(t_{m+1}) := 0$;
       end;
$H[e_6''$(no_data)] :
       $y :=$ no_data;
       $W(e_2''$(taim_$D$))$, t_{m+1})) := t_m + \xi_{2i}$;
       act_$D := 1$;
       end;
```

Aggregate Station_0. Functioning of this aggregate is similar that of aggregate Station nr and therefore only the differencess are presented in respect to aggregate Station nr.

1. Set of input signals:

$$X_0 = \{[X_{nr} \backslash \text{st\_on}] \cup \text{fail}(m)\},$$

where m – is the number stations switched on.

2. Set of output signals:

$$Y = \{[Y_{nr}\backslash \text{fail}] \cup \text{new\_st}(m)\},$$

where m – the number of the switched on station.

3. Set of internal events:

$$E_0'' = \{[E_{nr}''\backslash e_5''(\text{swit\_off})] \cup [e_7''(\text{taim\_}T),$$
$$e_{8i}''(\text{swit\_on}),\ i = \overline{1, n-1}]\},$$

where $e_7''(\text{taim\_}T)$ – end of taimer T; $e_{8i}''(\text{swit\_on})$ – i-th station switched on.

4. It is introduced controlling sequences for the events $e_7''(\ldots)$ and $e_{8i}''(\ldots)$:

$$e_7''(\text{taim\_}T) \to \{T\},$$
$$e_{8i}''(\text{swit\_on}) \to \{\xi_{8ij}\},\ i = \overline{1, n-1}, j = \overline{1, \infty},$$

where ξ_{8ij} – the operation duration after finishing the i-th station is switched on;
T – the duration of timer T.

5. $\nu(t_m) = \{\text{st}(t_m),\ \text{act\_}D(t_m)\}$.

6. Initial state

$$\text{act\_}D(t_0) := 1;$$
$$W(e_7''(\text{taim\_}T), t_0) := t_0 + T;$$
$$W(e_2''(\text{taim\_}D), t_0) := t_0.$$

7. Transfer operators:

```
$H[e_5'(\text{fail}(m))]$ :
    $W(e_{8,m-1}''(\text{swit\_on}), t_{m+1}) := t_m + \xi_{8i}$;
    end;
$H[e_7''(\text{taim\_}T)]$ :
    $W(e_1''(\text{taim\_}DI), t_{m+1}) := t_m + \tau_{DI}$;
    end;
```

where τ_{DI} – the duration of DI timer.

```
$H[e''_{8k}(\text{swit\_on})]$ :
    $y_4 := \text{new\_st}(k+1)$;
    **end**;
```

Aggregate Bus.

1. Set of input signals.

$$X = \{[\text{beg\_fr,end\_trans,no\_date,new\_st}(m)]_0, [\text{beg\_fr,end\_trans,no\_date,fail}]_1, \dots, [\text{beg\_fr,end\_trans,no\_date,fail}]_{n-1}\}.$$

2. Set of output signals

$$Y = \{[\text{fr\_end}(m), \text{bus\_is\_oc,no\_date}(m), \text{fail}(m)]_0, [\text{fr\_end}(m), \text{bus\_is\_oc,no\_date}(m), \text{st\_on}]_1, \dots, [\text{fr\_end}(m), \text{bus\_is\_oc,no\_date}(m), \text{st\_on}]_{n-1}\}.$$

3. Set of internal events.

$$E'' = \varnothing.$$

4. State

$$\nu(t_m) = \{q_i(t_m),\ i = \overline{1,n},\ \text{kan}(t_m)\},$$

where $q_i(t_m)\}\{1,2,\dots n\}$,
$q_i(t)$ – the number of successor for i-th station,

$$\text{kan}(t_m) = \begin{cases} 0, & \text{channel is idle;} \\ 1, & \text{channel is occupied.} \end{cases}$$

5. Initial state

```
**kan** $(t_0) := 0$;
$i = 1$; **while** $i < n$ **do begin**
    $q_i(t_0) := i + 1$;
    $i := i + 1$;
    **end**.
```

6. Transfer operators:

```
$H[e'_{1k}$ (new_st($p$))] : $k = \overline{2,n}$
    $i := p$;
    if $i = n$ then $i := 0$
    while $q_i(t_m) := 0$ do begin
        $i := i + 1$;
        if $i = n$ then $i := 0$
        end;
        $j := 1$;
        while $q_j(t_m) \neq i + 1$ do $j := j + 1$;
        $q_j(t_{m+1}) := p$;
        $q_p(t_{m+1}) := i + 1$;
        $y :=$ st_on;
        end;
$H[e'_{2k}$(beg_fr)] : $k = \overline{1,n}$
    kan $(t_{m+1}) := 1$;
    for $i := 1$ to $n$ do
        if $i \neq k$ and $q_i(t_m) > 0$ then
            $y_i :=$ bus_is_oc
    end;
$H[e'_{3k}$(end_trans)] : $k = \overline{1,n}$
    kan $(t_{m+1}) := 0$;
    for $i := 1$ to $n$ do
        if $i \neq k$ and $q_i(t_m) > 0$ then
            $y_i :=$ fr_end[$q_k(t_m)$]
    end;
$H[e'_{4k}$(no date)] : $k = \overline{1,n}$
    for $i := 1$ to $n$ do begin
        if $i \neq k$ and $q_i(t_m) > 0$ then
            $y_i :=$ no_date($q_k(t_m)$)
    end;
$H[e'_{5k}$(fail)] : $k = \overline{2,n}$
    $y_1 :=$ fail($k$);
    $i := 1$;
    while $q_i(t_m) \neq k$ do $i := i + 1$;
```

```
$q_i(t_{m+1}) := q_k(t_m)$;
$q_k(t_{m+1}) := 0$;
end;
```

3. Results of validation. Corretness created specification was investigeted by used protocol analysis system PRANAS-2 (Pranevitchius, Pilkauskas and Chmieliauskas, 1991). This system permits investigate general protocol properties such as: 1) completness, 2) deadlock freeness, 3) boundedness, 4) cyclic behaviour, 5) termination.

In Tables 1 – 3 some results of validation are represented. The numbers which are included in brackets ... means number of state. Numbers written after L, MO and $M[\_i]$ have the following meanings of discrete coordinates of state:

$$L : q_1\ q_2\ q_3\ \text{kan}$$

$$MO, nr, \text{act\_}D, \text{act\_}DI, \text{act\_}T, \text{act\_trans\_tr},$$
$$\text{st}, n\text{\_act}$$

$$M[\_i],\ i = \overline{2,3} : nr, \text{act\_}D, \text{act\_trans\_tr},$$
$$\text{act\_}DI, \text{sw}, \text{st}$$

In each state active operations are indicated also. Table 1 illiustrates altering discrete and continuous coordinates transmitting frames in stations $MO, M1$ and $M2$.

Table 2 illiustrates altering coordinates when the first station is switched off and switched on. Table 3 illiustrates switchings off and switchings on of the first and the second stations.

Table 1.

```
{32} L : 2 3 1 0 MO :    1 0 1 1 0 1 3 Tim_T Arr_Fr Taim_DI
               M[__i] :  2 0 0 0 1 1 Swit_of Arr_Fr
               M[__i] :  3 0 0 0 1 1 Swit_of Arr-Fr
                         ↓  Taim_DI in MO
{58} L : 2 3 1 1 MO :    1 0 0 1 1 1 3 Tim_T Arr_ Fr Trans_Fr
               M[__i] :  2 0 0 0 1 1 Swit_of Arr_Fr
```

```
                 M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                            ↓  Trans_Fr in MO
{104} L : 2 3 1 0 MO : 1 1 0 1 0 0 3 Tim_T Arr_ Fr Taim_D
                    M[___i] : 2 0 0 1 1 1 Swit_of Arr_Fr Taim_DI
                    M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                            ↓  Taim_DI in M1
 {79} L : 2 3 1 1 MO : 1 0 0 1 0 0 3 Tim_T Arr_Fr Taim_D
                    M[___i] : 2 0 1 0 1 1 Swit_of Arr_Fr Trans_Fr
                    M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                            ↓  Trans_Fr in M1
{150} L : 2 3 1 0 MO : 1 0 0 1 0 0 3 Tim_T Arr_Fr Taim_D
                    M[___i] : 2 1 0 0 1 0 Swit_of Arr_Fr Taim_D
                    M[___i] : 3 0 0 1 1 1 Swit_of Arr-Fr Taim_DI
                            ↓  Taim_DI in M1
 {92} L : 2 3 1 1 MO : 1 0 0 1 0 0 3 Tim_T Arr_Fr
                    M[___i] : 2 0 0 0 1 0 Swit_of Arr_Fr
                    M[___i] : 3 0 1 0 1 1 Swit_of Arr-Fr Trans_Fr
                            ↓  Trans_Fr in M2
{176} L : 2 3 1 0 MO : 1 0 1 1 0 0 3 Tim_T Arr_Fr Taim_DI
                    M[___i] : 2 0 0 0 1 0 Swit_of Arr_Fr
                    M[___i] : 3 1 0 0 1 0 Swit_of Arr-Fr Taim_DI
                            ↓  Taim_DI in M0
  {4} L : 2 3 1 1 MO : 1 1 0 1 0 0 3 Tim_T Arr_Fr Taim_D
                  M[___i] : 2 0 0 1 1 0 Swit_of Arr_Fr Taim_DI
                  M[___i] : 3 1 0 0 1 0 Swit_of Arr-Fr
                            ↓  Taim_DI in M1
 {17} L : 2 3 1 1 MO : 1 0 0 1 0 0 3 Tim_T Arr_Fr
                    M[___i] : 2 1 0 0 1 0 Swit_of Arr_Fr Taim_D
                    M[___i] : 3 0 0 1 1 0 Swit_of Arr-Fr Taim_DI
```

Table 2.

```
{32} L : 2 3 1 0 MO : 1 0 1 1 0 1 3 Tim_T Arr_Fr Taim_DI
                M[___i] : 2 0 0 0 1 1 Swit_of Arr_Fr
                M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                        ↓  Swit_off in M1
{30} L : 2 3 1 0 MO : 1 0 1 1 0 1 3 Tim_T Arr_Fr Taim_DI
                M[___i] : 2 0 0 0 0 0
                M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                        ↓  Taim_DI in M0
{56} L : 2 3 1 1 MO : 1 0 0 1 1 1 3 Tim_T Arr_Fr Trans_Fr
                M[___i] : 2 0 0 0 0 0
                M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                        ↓  Trans_fr in MO
{186} L : 3 0 1 0 MO : 1 0 1 1 0 0 2 Swit_on Tim_T Arr_Fr Taim_DI
                M[___i] : 2 0 0 0 0 0
                M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                        ↓  Swit_on in MO
{8} L : 2 3 1 0 MO : 1 0 1 1 0 0 3 Tim_T Arr_Fr Taim_DI
                M[___i] : 2 0 0 0 1 0 Swit_of Arr_Fr
                M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
```

Table 3.

```
{32} L : 2 3 1 0 MO : 1 0 1 1 0 1 3 Tim_T Arr_Fr Taim_DI
                M[___i] : 2 0 0 0 1 1 Swit_of Arr_Fr
                M[___i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                          ↓  Swit_off in M1
{30} L : 2 3 1 0 MO : 1 0 1 1 0 1 3 Tim_T Arr_Fr Taim_DI
                M[___i] : 2 0 0 0 0 0
```

```
                 M[__i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                            ↓  Taim_DI in MO
{56} L :  2 3 1 1 MO : 1 0 0 1 1 1 3 Tim_T Arr_Fr Trans_Fr
                 M[__i] : 2 0 0 0 0 0
                 M[__i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                            ↓  Trans_fr in MO
{102} L :  2 3 1 0 MO : 1 1 0 1 0 0 3 Tim_T Arr_Fr Taim_D
                 M[__i] : 2 0 0 0 0 0
                 M[__i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                            ↓  Taim_D in MO
{186} L :  3 0 1 0 MO : 1 0 1 1 0 0 2 Swit_on Tim_T Arr_Fr Taim_DI
                 M[__i] : 2 0 0 0 0 0
                 M[__i] : 3 0 0 0 1 1 Swit_of Arr-Fr
                            ↓  Swit_off  in M2
{185} L :  3 0 1 0 MO : 1 0 1 1 0 0 2 Swit_on Tim_T Arr_Fr Taim_DI
                 M[__i] : 2 0 0 0 0 0
                 M[__i] : 3 0 0 0 0 0
                            ↓  Arr_fr in MO
{318} L :  3 0 1 0 MO : 1 0 1 1 0 1 2 Swit_on Tim_T Arr_Fr Taim_DI
                 M[__i] : 2 0 0 0 0 0
                 M[__i] : 3 0 0 0 0 0
                            ↓  Taim_DI in MO
{192} L :  3 0 1 1 MO : 1 0 0 1 1 1 2 Swit_on Tim_T Arr_Fr Trans_Fr
                 M[__i] : 2 0 0 0 0 0
                 M[__i] : 3 0 0 0 0 0
                            ↓  Trans_fr in MO
{326} L :  3 0 1 0 MO : 1 1 0 1 0 0 2 Swit_on Tim_T Arr_Fr Taim_D
                 M[__i] : 2 0 0 0 0 0
                 M[__i] : 3 0 0 0 0 0
```

$\downarrow$ **Taim_*D* in *MO***

{506} L: 1 0 0 0 MO: 1 0 1 1 0 0 2 Swit_on Swit_on Tim_*T* Arr_Fr Taim_*DI*

$M[\_\_i]$: 2 0 0 0 0 0

$M[\_\_i]$: 3 0 0 0 0 0

$\downarrow$ **Arr_fr in *MO***

{722} L: 1 0 0 0 MO: 1 0 1 1 0 1 1 Swit_on Swit_on Tim_*T* Arr_Fr Taim_*DI*

$M[\_\_i]$: 2 0 0 0 0 0

$M[\_\_i]$: 3 0 0 0 0 0

$\downarrow$ Swit_on [1]

{524} L: 2 1 0 0 MO: 1 0 1 1 0 1 2 Swit_on Tim_*T* Arr_Fr Taim$_D I$

$M[\_\_i]$: 2 0 0 0 0 0 Swit_of Arr_Fr

$M[\_\_i]$: 3 0 0 0 0 0

$\downarrow$ Swit_on [2]

{3} L: 2 3 1 0 MO: 1 0 1 1 0 1 3 Tim_*T* Arr_Fr Taim_*DI*

$M[\_\_i]$: 2 0 0 0 0 1 0 Swit_of Arr_Fr

$M[\_\_i]$: 3 0 0 0 0 0 Swit_of Arr-Fr

4. Simulation results. Simulation results are represented in Table 4. Parameters of model are the following:

Taim_Frame – duration of frames,

Taim_Head – duration of head of frames,

Taim_*D* – duration of taimer *D*,

Taim_*DI* – duration of taimer *DI*,

Taim_*T* – duration of taimer *D*,

V – velocity of chanel,

n – number of station,

Arr_Frame – parameter of puasonian input stream,

*T*_swit_on and *T*_swit_off – intensivity of operations swit_on and swit_off, which have exponential distributions.

Characteristics of model:

Table 4. Simulation results. Tau_Frame=800 bit, Tau_Head=160 bit, Tau_NoData=48 bit, Taim_D=0,0000025 s, Taim_DI= 0,0000012 s, Taim_T=100 s, Swit_on=swit_off= 8

1. $V = 10000000$, Arr_Frame $= 0,001$ 1/s

n	T_Wait	L_Wait	R_Useful	K_Full
2	0,00011	0,00001	0,1418	0,8323
4	0,00013	0,00003	0,2806	0,8639
6	0,00016	0,00006	0,4118	0,8939
8	0,00019	0,00010	0,5297	0,9207
10	0,00025	0,00016	0,6304	0,9437
2. $V = 50000000$ b/s, Arr_Frame $= 0,001$ 1/s				
2	0,00002	0,00000	0,0307	0,4639
4	0,00002	0,00001	0,0611	0,4831
6	0,00003	0,00003	0,0917	0,5025
8	0,00003	0,00001	0,1219	0,5217
10	0,00003	0,00001	0,1529	0,5413
3. $V = 50000000$ b/s, Arr_Frame $= 0,000135135$ 1/s				
2	0,00002	0,00000	0,2012	0,5921
4	0,00003	0,00001	0,3848	0,6882
6	0,00004	0,00002	0,5399	0,7864
8	0,00006	0,00004	0,6513	0,8569
10	0,00009	0,00007	0,7160	0,8979

T_Wait – mean value of transmitting of frame including waiting time,

L_Wait – mean value of waiting time,

K_Useful – coefficient useful using of channel,

K_Full – cofficient of full using channel.

5. Conclusions. Formal analysis of protocol permitted to check correctness of functioning of protocol in diffucult situations. The aggregate method and the protocol analysis tools were found to be well suited for event controlled protocols. The main advantage of this method is that the same specification is used for creating both validation and simulation models. The protocol was found to be correct. Simulation results can be used to set the parameters of the protocol to meet the requirements of a particular application.

REFERENCES

Pranevitchius, H. (1991). Aggregate approach for specification, validation, simulation and implementation of computer network protocols. In *Lectures Notes in Computer Science*, No 502, Springer-Verlag, pp. 431–437.

Pranevitchius, H., V.Pilkauskas, and A.Chmieliauskas (1991). Automatizated system for validation and simulation computer network protocols PRANAS-2. In *Automatic and Computer Technic*, No 6, Riga, pp. 9–18.

Sintonen L. (1990). Event Driven Bus Architecture for Bounded Area Networks. In *Proc. 16 th Annual Conference of IEEE Industrial Electronics Sciety*, Pacific Grive CA, USA, pp. 539–541.

Received February 1993

H. Pranevitchius received the Degrees of Candidate of Technical Sciences at the Kaunas Politechnic Institute, Kaunas, Lithuania and Computer Science Institute of Latvian Academy of Sciences, Riga, Latvija, in 1970 and 1984, respectively. He is currently Professor and head the Departament Managment Informatic, Kaunas University of Technology. His research interests include simulation of complex systems and specification, validation, testing and simulation of computer networks protocols.

L. Sintonen Dr.tech., currently associate professor in Tampere University of Technology, Software Systems Laboratory. Graduated as diploma engineer in microwave and radio engineering from Helsinki University of Technology in 1966, licenciate 1971 and doctor 1980 in computer science, both from Tampere University of Technology. He has been working in industry, and since 1968 in Tampere University of Technology, first as laboratory engineer, later as associate professor since 1971. The main areas of interest are computer networks, high speed networks and distributed computer systems. Current research topics are.protocol design and performance analysis and communication in distributed systems.