INFORMATICA, 1990, Vol.1, No.1, 107–124

CORRECTNESS ANALYSIS AND SIMULATION OF COMPUTER NETWORK PROTOCOLS BY AGGREGATE APPROACH

Henrikas PRANEVITCHIUS

Control system Department,
Kaunas Polytechnic Institute,
233000 Kaunas, V.Juro St.50, Lithuania

Abstract. The aggregate approach to the formal description, verification and simulation of computer network protocols is considered in the paper. With this approach, the offered design stages can be performed using a single mathematical scheme. The reachability analysis method and the program proof technique are viewed as methods for correctness analysis. The proposed approach for correctness analysis and model construction was used in creating the protocol analysis system PRANAS.

Key words: protocol correctness analysis, simulation, aggregate approach.

Introduction. A design of computer network protocols involves the following steps: formal description, verification, simulation and program implementation.The formal methods described in numerous papers are appropriate only to some steps of design, this being their main disadvantage in designing computer network protocols. Therefore,preference is given to such formal methods,which provide the basis for creating the techniques for simulation and analysis of the protocol correctness.

The aggregate mathematical schemes for a formal description of complex systems are presented by Pranevitchius (1982). Their application facilitates the creation of the construction for simulation models and analysis of the correctness of the models being created.

In the aggregate approach the system being modelled is viewed as a whole of interacting aggregates. The author's method of control sequences is used to describe every aggregate. In the aggregate description the following sets should be described: sets of input and output signals, a set of states and output signals, a set of states, transition and output operators.

With the formal description methods of the functioning system, the principles for automatic construction of aggregate simulation models were developed.

The reachability analysis and program proof technique are used for the analysis of the correctness of the models. The reachability analysis is based on the concept of the global state, which is achieved by the composition of the aggregative model of a protocol. Analyzing the reachability graph, the following properties of the model are observed: 1) completeness, 2) deadlock freeness, 3) tempoblocking freeness, 4) liveliness, 5) termination or cyclic behaviour, 6) boundedness, etc.

The invariant approach is used to analyze the correctness of the aggregative models by the program proof technique. The invariant is an assertion, describing correct system functioning and remaining true, despite the events taking place and transition from one state of the system to another. The trueness of the invariant should be proved for every fragment of a model related to the event. The correspondence between the aggregative model and the conceptual model can be checked by this method.

The above approaches were used to create the system PRANAS (Pranevitchius, Chmieliauskas and Pilkauskas, 1985), (PRotocol ANAlyzing System), which is used for specification, verification and simulation of computer network protocols. With this system, the construction of the aggregative simulation models is automated and the correctness of the model is examined by the reachability analysis method.

An example of a formal description of some control procedure for computer network protocols is given below.

Aggregate approach and method of controlling sequences for protocol formalization. Many different mathematical schemes, including the aggregate ones, are used to formalize computer network protocols an advantage of the aggregate mathematical schemes. However, for other approaches it is in that the models for the analysis of both correctness and performance can be constructed using a single protocol specification (Pranevitchius and Chmieliauskas, 1983). In this case the protocol is considered to be a set of interacting piece-linear aggregates. An aggregate is a system with N input and M output contacts. Signals $X = \{x_1, x_2, ..., x_N\}$ arrive at the input contacts, while signals $Y = \{y_1, y_2, ..., y_M\}$ are obtained from the output contacts. The aggregate functioning is examined on a set of time moments $T = \{t_0, t_1, ..., t_m, ...\}$, at which one or several events take place, resulting in the aggregate state alteration. The set of events, which may take place in the aggregate is denoted as $E.E = E' \bigcup E'', E' \bigcap E'' = \varnothing$, in which $E' = \{e'_1, e'_2, ..., e'_N\}$ is a set of external events, i.e., the arrival of input signals from the set X. $E'' = \{e''_1, e''_2, ..., e''_f\}$ is a set of internal events and f is a number of operations taking place in the aggregate. The events in the set E indicate the end of the operations. Every internal event has a controlling sequence $\{\xi^{(j)}_i\}, j = \overline{1, \infty}$, in which $\xi^{(1)}_i$ is duration of the operation followed by the event e''_1.

The aggregate state is determined as follows:

$$Z(t_m) = [\nu(t_m),\ z_\nu(t_m)],$$

in which $\nu(t_m) = \{\nu_{01}(t_m), \nu_{02}(t_m), \ldots, \nu_{0p}(t_m)\}$ is a discrete component. The coordinates of the discrete component determine the state of the elements in the system, $Z_\nu(t_m) = \{W(e''_1, t_m), W(e''_2, t_m), \ldots, W(e''_f, t_m)\}$ are the controlling coordinates determining the moments of the occurrence of internal events. For every i, $W(e''_i, t_m)$ has a value ∞, if at the moment t_m the operation does not take place, otherwise the value $W(e''_i, t_m)$ is determined by a value of the element of the appropriate controlling sequence.

Transition from one aggregate state to another results from the occurrence of an event. The algorithm of the aggregate of the change in the coordinates of the state $Z(t_m)$ during the transition to the state $Z(t_{m+1})$, $m = 0, 1, \ldots$, and by the duration of the intervals $\Delta t_m = t_{m+1} - t_m$ between the two events. The aggregate evolution may be characterized by a sequential change in the moments t_m and in the respective events e_m, i.e.,

$$(t_0, e_0), (t_1, e_1), \ldots, (t_m, e_m), \ldots, t_0 \leqslant t_1 \leqslant \ldots \leqslant t_m \leqslant \ldots,$$

where t_0 is the starting moment.

In the state $Z(t_m)$, the moment t_{m+1} of the occurrence of a next event is determined by the moment of the arrival of an input signal or by the equation

$$t_{m+1} = \min_{1 \leqslant i \leqslant f} W(e''_1, t_m).$$

A new aggregate state is determined by the transition operator H:

$$Z(t_{m+1}) = H[Z(t_m), e_i], \qquad e_i \in E' \bigcup E''.$$

The output signals y_i, from the set of output signals $Y = \{y_1, y_2, \ldots, y_M\}$, may be sent by the aggregate only at the moments of the occurrence of events from the subset E' and E''. The content of the output signals is determined by the output operator G:

$$y = G[Z(t_m), e_i], \qquad e_i \in E' \bigcup E'', \qquad y \in Y.$$

A more detailed description of the aggregate method, in which the controlling sequences are used is given by Pranevitchius (1982).

Assume that the protocol being modelled is described by the aggregate system consisting of K aggregates. Assume that the k-th aggregate contains N_k inputs and M_k outputs. The total number of the communication channels that transmit signals between the aggregates is denoted L. The matrices R and H are used to combine the aggregates into an aggregate system. The matrix R determines the aggregate and the number of the pole, to which the output signal is transmitted through the particular channel. $R = ||r_{ji}||$, $i = \overline{1, L}$, $j = \overline{1, 2}$, where r_{ji} is the number of the aggregate, which receives the input signal from the i-th communication channel, $1 \leqslant r_{1i} \leqslant K$, r_{21} is the number of the input pole of the aggregate r_{1i}, which is incidental to the i-th channel, $1 \leqslant r_{21} \leqslant Nr_{1i}$. The matrix H directs every signal from the set Y to a channel through which the output signal is to be transmitted. $H = ||h_{ij}||$, $i = \overline{1, K}$, $j = \overline{1, max\{M_K\}}$, where h_{ij} is the number of the channel incidental to the i-th output pole of the i-th aggregate $1 \leqslant h_{ij} \leqslant L$.

The set of the events which may occur in the aggregate system is the union of the subsets of the internal events, which take place in each aggregate, i.e.,

$$E = \bigcup_{k=1}^{K} E_k,$$

in which E_k is the set of the events, taking place in the k-th aggregate.

The next moment of time t_{m+1}, at which an event from the set E will occur, is determined as follows:

$$t_{m+1} = \min_{1 \leqslant k \leqslant K} (\min_{e_r'' \in E_K} W_k(e_r'', t_m)).$$

Aggregative model of the basic procedure of data transmission. An object A transmits protocol elements INFO (information) and REQ (request). An object B transmits protocol elements ACK (a response to INFO) and RES (a response to REQ). The exchange of the protocol elements takes place in the duplex channel. The INFO transmitted are sequentially numbered. The protocol elements ACK and RES contain the number of the INFO acknowledged.

INFO transmission is timer controlled. The timer is started on the INFO transmission and stopped on receiving the acknowledgement. The reset takes place after the timer expiration, provided no acknowledgement is received after the INFO transmission (single or several). The object A transmits REQ on reset. After the transmission the timer is started which stops on receiving the response RES to REQ. The reset having been completed, the change to normal (standard) data transmission takes place. The object A contains a meter (of the INFO transmitted) for the control of the sequence of the INFO transmitted, and the object B - a meter of the INFO received. The flow control is based on the "window" principle.

The objects A and B can be represented as two separate interacting aggregates. The aggregate A is meant for transmission of the protocol elements INFO and REQ as well as for receiving the protocol elements ACK and RES, acknowledging the receipt of INFO and REQ. The aggregate B is meant for receiving the protocol elements INFO and REQ. It also transmits the acknowledging protocol elements ACK and RES.

Aggregate A

1. A set of input signals:

$$X = \{x_1, x_2\},$$

where $x_1 = i$ – the protocol element ACK numbered i has arrived.

2. A set of external events:

$$E = \{e'_1, e'_2\},$$

where e'_1 – the input signal x_1 has arrived,
e'_2 – the input signal x_2 has arrived.

3. A set of output signals:

$$Y = \{y_1, y_2\},$$

where $y_1 = i$ – the protocol element INFO numbered i is being transmitted,
y_2 – the protocol element REQ is being transmitted.

4. A set of internal events:

$$E'' = \{e''_1, e''_2, e''_3, e''_4\},$$

where e''_1 – the formation of INFO is completed,
e''_2 – the timer has expired and REQ has been formed,
e''_3 – INFO transmission has been completed,
e''_4 – REQ transmission is completed.

5. State vector:

$$Z_1(t) = \{v_1(t), W_1(t)\},$$

where $W_1(t)$ – a continuous state component:
$$W_1(t) = \{w(e_1), t), \ i = \overline{1,4}\},$$
where the coordinates $w(e''_1, t)$, $i = \overline{1,4}$ can either have the final values or be undetermined. If the continuous coordinate

is undetermined, the corresponding events e_i'', $i = \overline{1,4}$ cannot take place at a given state of the aggregate.

$v_1(t)$ – a discrete state component,

$v_1(t)$ = {varsend(t), lastnr(t), char A(t)},

where varsend(t) – the number of INFO transmitted,

lastnr(t) – the number of INFO acknowledged,

char A(t) – the channel state character from the aggregate A to the aggregate B, acquiring the meanings:

$$char\ A(t) = \begin{cases} 0, & \text{the channel is free,} \\ 1, & \text{INFO is being transmitted,} \\ 2, & \text{REQ is bėing transmitted.} \end{cases}$$

6. Transfer and output operators:

The description of transfer and output operators is given below:

```
$H(e_1')$ : if $lastnr(t_m), i \leqslant varsend(t_m)$then do;
        $lastnr(t_{m+1}) = i;$
        $w(e_1'', t_{m+1}) < \infty;$
    if $i = varsend(t_m)$then $w(e_2'', t_{m+1}) = \infty;$
    else $w(e_2'', t_{m+1}) < \infty$; end;
        $G(e_1') :\ Y = 0;$

$H(e_2')$:if $lastnr(t_m) \leqslant i \leqslant varsend(t_m)$ then do;
    $varsend(t_{m+1}) = lastnr(t_m);$
        $W(e_1'', t_{m+1}) < \infty;$
        $W(e_2'', t_{m+1}) = \infty;$
        end;
    $G(e'_2): \ Y = 0;$
$H(e''_1)$; if $varsend(t_m) < lastnr(t_m) + wind$
    then do;

        $char\ A(t_{m+1}) = 1;$
        $nssend = varsend(t_m);$
        $varsend(t_{m+1}) = varsend(t_m) + 1;$
        $W(e''_3, t_{m+1}) < \infty;$
        if $\mathrm{w}(\mathrm{e}''_2, \mathrm{t_m}) = \infty$ then $\mathrm{w}(\mathrm{e}''_2, \mathrm{t_{m+1}}) < \infty;$
    end;
    else $\mathrm{w}(\mathrm{e}''_1, \mathrm{t_{m+1}}) = \infty;$
    $G(e''_1): \ Y = 0;$
$H(e''_2): \ char\ A(t_{m+1}) = 2;$
    $W(e''_1, t_{m+1}) = \infty;$
    $w(e''_2, t_{m+1}) < \infty;$
    $G(e''_2): \ Y = 0;$
$H(e''_3): \ char\ A(t_{m+1}) = 0;$
$H(e''_4): \ char\ A(t_{m+1}) = 0;$
    $w(e''_3, t_{m+1}) = \infty;$
    $w(e''_4, t_{m+1}) = \infty;$
    $G(e''_4): y_2 = (\mathrm{REQ}).$
```

Aggregate B.

1. A set of input signals:

$$X = \{x_1, x_2\},$$

where $x_1 = i$ – the protocol element INFO numbered i has arrived,

x_2 – the protocol element REQ has arrived.

2. A set of external events:

$$E = \{e'_1, e'_2\},$$

where e'_1 – the input signal x_1 has arrived and the protocol element ACK has been formed,

e'_2 – the signal x_2 has arrived and the protocol element RES has been formed.

3. A set of output signals:

$$Y = \{y_1, y_2\},$$

where $y_1 = i$ – the protocol element *ACK* numbered i is being transmitted,

$y_2 = i$ – the protocol element *RES* numbered i is being transmitted.

4. A set of internal events:

$$E'' = \{e''_5, e''_6\},$$

where e''_5 – the transmission of the protocol element ACK has been completed,

e''_6 – the transmission of the protocol element RES has been completed.

5. State vector:

$$Z_2(t) = \{v_2(t), W_2(t)\},$$

where $W_2(t)$ – a continuous state component,

$$W_2(t) = \{w(e''_i, i = \overline{5,6}\},$$

where the coordinates $w(e''_i, t), i = \overline{5,6}$ can have either the finite values or be undetermined,
$v_2(t)$ – a discrete state component,

$$v_2(t) = \{varrcv(t), charB(t)\},$$

$varrcv(t)$ – the number of the INFO received,
$charB(t)$ – the channel state character from the aggregate A to the aggregate B, acquiring the meanings:

$$char\ B(t) = \begin{cases} 0, & \text{the channel is free,} \\ 1, & \text{ACK is being transmitted,} \\ 2, & \text{RES is being transmitted.} \end{cases}$$

6. Transfer and output operators:

$$H(e'_1) : charB(t_{m+1}) = 1;$$

$$if\ i = varrcv(t_m) \text{ then } varrcv(t_{m+1}) = varrcv(t_m) + 1;$$

$$W(e''_5, t_{m+1}) < \infty;$$

$$G(e'_1) : Y = 0;$$

$$H(e'_2) : charB(t_{m+1}) = 2;$$

$$w(e''_5, t_{m+1}) = \infty;$$

$$w(e''_6, t_{m+1}) < \infty;$$

$$G(e'_2) :\ Y = 0;$$

$$H(e''_5) : charB(t_{m+1}) = 0;$$

$$w(e''_5, t_{m+1}) = \infty;$$

$$G(e_6''): charB(t_{m+1}) = 0;$$

$$w(e_6, t_{m-1}) = \infty;$$

$$G(e_6''): y_2 = (varrcv(t_m)).$$

Verification and simulation. The above presented approach of the formal protocol description facilitates the creation of simulation models. On the basis of this approach some software systems have been developed. The systems SAPAS(Gorelik and Pranevitchius, 1985) and SIMAS(Pranevitchius and Janilionis, 1985) facilitate automatic construction of aggregate simulation models. These systems facilitate the automation of stages in creating the program models of individual aggregates and aggregate systems and making simulation experiments.

The formal aggregate description of protocols may be used for the creation of a reachability graph, analyzing the protocol correctness. The nodes of the reachability graph determine the set of states, in which the protocol may be present, while the arcs indicate a possible transition from one state to another. Each node of the graph determines the global protocol state, i.e., the state which is reached after the composition of all the aggregates of an aggregate system. The global protocol state is as follows:

$$Z(t_m) = \bigcup_{i=1}^{M} (\nu_i(t_m), z_{\nu_i}(t_m)),$$

in which M is the number of aggregates in the aggregate system.

The formal aggregate description used in the construction of simulation models is sufficient for creating the reachability graph and there is no need to make use of controlling sequences. The continuous coordinates of aggregate states have only two values in this case, i.e., $W(e''_{ij}, t_m) = \infty$,if the event e''_{ij} cannot occur, and $\neq \infty$, if the event may occur.

The set of the events which may occur in the state $Z(t_m)$ is determined by the set of continuous coordinates, for which $W(e''_{ij}, t_m) \neq \infty$. Accordingly, the set of adjacent states is determined as follows:

$$D^*[Z(t_m)] = \{Z(t_{m+1}) : Z(t_{m+1}) = H(Z(t_m), e''_{ij}),$$
$$\forall ij,\ W(e''_{ij}, t_m) \neq \infty\}.$$

It can be concluded from the above, that the aggregate approach can be used to a formal description, correctness analysis and simulation of computer network protocols. Its main advantage is that this approach makes it possible to create models for the analysis of both protocol correctness and performance, using a single mathematical description scheme in the form of interacting piece-linear aggregates. This approach was used in creating the protocol analysis system PRANAS (Pranevitchius, Chmieliauskas and Pilkauskas, 1985), which facilitates the protocol verification and simulation,using a single specification.

The main components of the PRANAS system are:

1) the protocol specification language AGGREGATE 84;
2) a preprocessor;
3) a verification subsystem;
4) a simulation subsystem.

The interacting protocol entities are represented by the piece-linear aggregates, whose interconnection schemes are given in the tables. The transmission medium may also be represented by an individual aggregate.The aggregates are described by the AGGREGATE 84 language.

The system of automatic analysis of the protocol correctness,described by the aggregate method,has been used for the correctness analysis of the basic procedure of data transmission.

The analysis carried out helped to detect the errors, emerging on passing from the conceptual model to the formal one. Fig 1 presents a fragment of the reachability states graph,which shows that the protocol can get into a deadlock cycle. The deadlock situation is the result of the fact that in the formal model the element RES has no function of the acknowledging the receipt of INFO elements.

The protocol specification, based on the aggregate approach, can be applied in the design of the simulation models for protocol efficiency analysis. For this,the following changes must be done in the specification. For every class of the aggregate model,the events control sequences are a set of the elements,which determine the duration of operation resulting in events. In the given model this can be represented by the following:

$$\{e''_{ij}\} \rightarrow \{\xi_{ij}\},\ i = \overline{1,6},\ j = 1,2,$$

where j – the number of the event e_i, occurring after the initial moment of simulation, ξ_{ij}–the duration of the operation followed by the event e''_{ij}.

The continuous coordinates of the states of the aggregate model are determinated as follows:

$$w(e''_i, t_{m+1}) = \begin{cases} t_{syst} + \\ +\xi_{i,r(e''_i,t_m)+1}, & \text{if at the system moment of time } t_{syst} \text{ an operation begins the completion of which is followed by an event } e''_i, \\ \infty, & \text{otherwise} \end{cases}$$

$r(e''_i, t_m)$ – the number of events taking place in the time interval $[0, t_m]$. For the simulation model to take into account the loss of protocol elements, the following changes are done in the output operators:

$$G(e''_i):\ \text{if } \zeta > \text{P then Y} = (\text{N}),\ \text{else Y} = 0,$$

$$\begin{tikzcd}
 & 00000\ W_1 \arrow[d, "e_1"] & & \\
 & 10100\ W_1W_2W_3 \arrow[d, "e_3"] & & \\
 & 10011\ W_1W_2W_5 \arrow[d, "e_1"] & & \\
 & 20111\ W_2W_3W_5 \arrow[d, "e_3"] & & \\
 & 20021\ W_2W_5 \arrow[d, "e_2"] & & \\
 & 20221\ W_2W_4W_5 \arrow[d, "e_4"] & & \\
 & 20022\ W_2W_5W_6 \arrow[d, "e_6"] & & \\
 & 00020\ W_1 \arrow[d, "e_1"] & & \\
 & 10120\ W_1W_2W_5 \arrow[d, "e_2"] & & \\
 & 10220\ W_2W_3W_4 \arrow[loop left, "e_2"] \arrow[dl, "e_3"'] \arrow[dr, "e_4"] & & \\
10221\ W_2W_4W_5 \arrow[loop left, "e_2"] \arrow[d, "e_5"'] & & 10022\ W_2W_6 \arrow[d, shift right, "e_2"'] \arrow[dr, "e_6"] & \\
10220\ W_2W_4 \arrow[loop left, "e_2"] \arrow[loop below, "e_6"] \arrow[urr, "e_4"] & & 10222\ W_2W_4W_6 \arrow[u, shift right, "e_4"'] \arrow[loop below, "e_2"] & 10020\ W_2 \arrow[lll, bend left, "e_2"]
\end{tikzcd}$$

Fig.1. Fragments off the reachable states graph

where e_i'' – the type of the internal event, N – the contents of the output signal, ζ – a random value randomly distributed in the interval $[0, 1]$, P – the protocol element loss probability.

Fig.2 shows the dependence of the general loading coefficient of the channel $k_{03}(1)$ and the useful loading coefficient of the channel $k_{n3}(2)$ on the relation τ/T, where τ – the mean time of signal transmission by the channel, T – the timer duration.

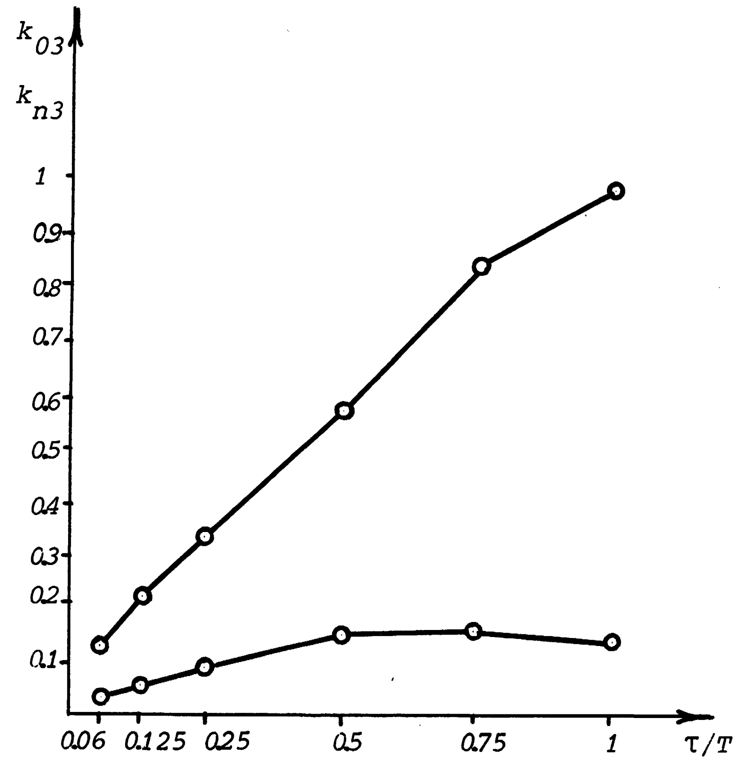

Fig.2. Simulation results

Using the reachability analysis for the protocol verification only some protocol properties are examined, i.e., boundedness, absence of overspecification, completeness, static and dynamic deadlock freeness, termination, etc. Conclusions about an error freeness in the protocol may be made by this approach for some error types only. The correspondence between the formal protocol description and its conceptual model cannot be determined by this approach. Some results in this respect can be achieved using the program proof technique. The methods for analyzing an aggregative model correctness by program proofs are presented by Pranevitchius and Panevėžys (1985). The gist of the methods is that we form an invariant system,which represents the correct system functioning and is to remain true in spite of the events and transitions taking place. The invariant is a logical assertion relation to the aggregative model coordinates.Since alterations in the aggregative model may take place only at the moment of the event occurrence, the invariant trueness must be proved in relation to each event in the aggregative model. A further development of the approach may be achieved through the automated proof.

Conclusions. Possibilities to apply the aggregate approach to formalization, verification and simulation of computer network protocols are surveyed in the paper. The main advantage of the approach is that it facilitates a model creation for both correctness analysis of the protocols and performance analysis introducing insignificant changes in the basic specification of the protocol. The approach may also be used for automated implementation of the protocols. Thus, all steps of the computer network protocol design, from specification to implementation, may be solved by the aggregate approach.

REFERENCES

Gorelik, Y.,and H.Pranevitchius (1985). Automated implementation system for simulation models. In *Teorija Modelirovanija Sloznych Sistem,* Acad. Sci. USSR, Moscow. pp. 83–92 (in Russian).

Pranevitchius, H. (1982). *Models and Methods for Computer System Investigation.* Mokslas, Vilnius. 228pp. (in Russian).

Pranevitchius, H.,and A.Chmieliauskas (1983). *Correctness Proof and Performance Predication of Protocols Using Aggregate Approach and Control Sequence Method.* Acad. Sci. USSR, Moscow. 32pp. (in Russian).

Pranevitchius, H., A.Chmieliauskas and V.Pilkauskas (1985). Protocol simulation and verification in PRANAS. In *Seti i Komutacija Paketov,* ESTI, Riga. pp. 209–213 (in Russian).

Pranevitchius, H.,and V.Janilionis (1985). Aggregative simulation sistem (SIMAS). In *Perspektivy Razvitija Vychisliteljnych Sistem,* RPI, Riga. pp. 147–149 (in Russian).

Pranevitchius, H.,and A.Panevėžys (1985). Analysis of aggregative models by assertion proofs. *Matematika i Matematicheskoe Modelirovanije.* KPI, Vilnius. pp. 32–34 (in Russian).

Received October 1989

H. Pranevitchius received the Degrees of Candidate and Doctor of Technical Sciences from the Kaunas Polytechnic Institute, Kaunas, Lithuania, and Computer Science Institute of Latvian Academy of Sciences, Riga, Latvija, in 1970 and 1984, respectively. He is a professor and heads the Department of Control Systems, Kaunas Polytechnic Institute. His research interests include simulation of complex systems and specification, validation and simulation of computer networks protocols.