

# The Use of Aggregate Approach for Formal Specification and Simulation of Real-Time Systems\*

**Henrikas Pranevicius and Dalius Makackas**

Business Informatics Department, Kaunas University of Technology  
Kaunas, Lithuania

E-mail: {hepran, d.makackas}@if.ktu.lt

## Abstract

Paper presents model of Control Area Network (CAN) protocol. The model was created using piece-linear aggregate formalism. CAN protocol is a component part of Anti-lock Braking System (ABS). The goal of performed investigations was to analyze protocol application possibilities in ABS. Created simulation model of CAN protocol permitted to evaluate waiting times of transmitted information packets. Requirements for this parameter are defined by ABS dynamical characteristics.

**Keywords:** CAN protocol, aggregate approach, real-time systems, simulation, formal specification.

## 1. Introduction

The stage of formal specification creation is one of the most important during the design of the software for real-time systems. Such formal specification usually is used for analysis and implementation purposes. During analysis stage it is necessary to resolve two tasks: analysis of logical correctness and evaluation of the systems functioning timing parameters.

Different mathematical schemes are used for creation of systems formal description, such as forms of automata models, Petri-nets, data flow and state transition diagrams, temporal logic of actions technique, abstract communicating methods, Quirk methodology [3], mixed and others.

While choosing a formalization method it is preferable to resolve both fore mentioned analysis tasks on the base of a single formal specification. Analysis task can be resolved on the base of a single formal description. Piece-linear aggregate approach or simply aggregate approach [7] has such property—it can be successfully used both for correctness analysis [5] and for evaluation of system functioning timing parameters. Specification language ESTELLE/Ag and specification analysis tool PRANAS-2 [6] were created on the base of Aggregate method. There are some differences between ESTELLE/Ag and the ESTELLE ISO standard: a model of piece-linear aggregate is used in ESTELLE/Ag (abbreviation "Ag" denotes Aggregate approach). The use of such a model instead of a finite-state automaton, which is the formal background of the standard ESTELLE, enables to create a single model both for validation and simulation. This is possible due to the special structure of the piece-linear aggregate. Apart from the discrete components describing the state of the modules of specified system, there are also continuous components to control event sequences in the module. These continuous components are called operations. Sequences of actions are described by means of operators. Intermediate results of operators are invisible to the outside. If such an operation sequence is being performed at a given instance of time the corresponding operation is called "active". Thus, an individual module evolves two types of events: arrival of input signal and completion of an active operation. Specification analysis system PRANAS-2 consists of the following software tools: specification editor, validation subsystem and simulation subsystem. Specification editor provides a capability

---

\* Investigations have been supported by COPERNICUS programme funded LIMITS project.

to create initial specifications in ESTELLE/Ag language. The validation subsystem permits to construct a validation model for the program by generating reachability graph. After construction of the reachability graph the following specification characteristics may be verified: completeness, deadlock freeness, boundedness, absence of static and dynamic deadlocks and termination.

An initial specification generated by specification editor has to be supplemented with additional code. This is necessary in order to define the duration of operations and to introduce additional variables for gathering statistics about evaluated system parameters.

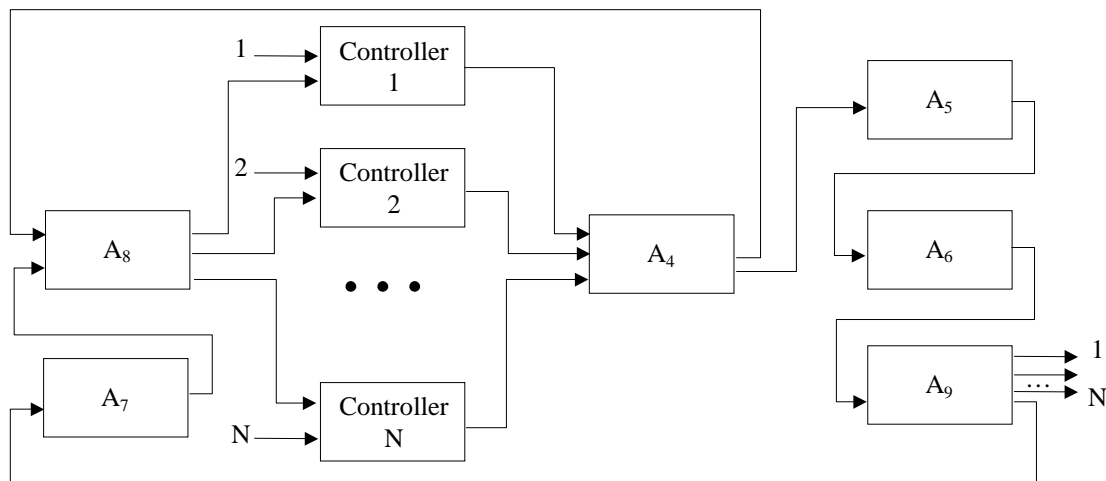
The possibility of using the aggregate approach for analysis of real time systems is illustrated by creation of an aggregate model of Control Area Network (CAN) protocol when this protocol is used in Anti-lock Braking System (ABS). The conceptual and functional descriptions and timing parameters of that system are taken from [9].

The goal of this investigation was to analyze CAN protocol application possibilities in ABS. CAN interface has to provide fast transmission of information packets. ABS forms requirements for high-speed transmission of information packets through CAN interface. Created CAN protocol simulation model was used for check of this requirement.

The paper is structured in the following way. Section 2 gives formal specification of CAN protocol using piece-linear aggregate formalism. Simulation results showing waiting times for packets with different priorities are presented in section 3. The last section presents requirements for transmission time of information packets through CAN interface, there it is also discussed how these requirements are fulfilled.

## 2. Aggregate specification of CAN protocol

The conceptual model of the CAN protocol is prepared using ISO standard [1], PHILIPS[4] and SIEMENS [8] documents. Figure 1 presents aggregate system interconnection scheme using these standards.



A<sub>4</sub> - the sending of the 1<sup>st</sup> bit of a message; A<sub>5</sub> - an arbitration procedure; A<sub>6</sub> - the transmission of informational part of a message; A<sub>7</sub> - an intermission between the transmission of messages; A<sub>8</sub> - a bus; A<sub>9</sub> - the transmission of control information of message.

Figure 1. Aggregate system interconnection scheme of CAN protocol

**Aggregate A<sub>4</sub>.** (The aggregate describes the sending of the 1<sup>st</sup> bit of a message).

1. The set of input signals:

$\{x_1, \dots, x_N\}$ , where  $x_i$  – signal from i-th controller.

2. The set output signals:  
 $\{y_1, y_2\}$ , where  $y_1$  – signal denoting start of transmission,  $y_2$  – signal denoting start of arbitration procedure.
3. The set of external events:  
 $\{e'_1, \dots, e'_N\}$ , where  $e'_i$  – means that  $i$ -th controller starts to send packet of  $l$ -th priority.
4. The set of internal events:  
 $\{e''_1, e''_2\}$ , where  $e''_1$  – means that transmission of start of frame bit has ended;  $e''_2$  – means that initialization of packet transmission has ended.
5. Controlling sequence:  
 $e''_1 \rightarrow \{\eta_i\}_{i=1}^{\infty}$ , where  $\eta_i = 1$ ,  $e''_2 \rightarrow \{\xi_i\}_{i=1}^{\infty}$  where  $\xi_i = 0.2$ .
6. The set of discrete components:  
 $z(t_m) = \{\chi_1(t_m), \chi_2(t_m)\}$ ,  
where  $\chi_1(t_m) = \begin{cases} 0, & \text{bus is empty;} \\ l, & \text{bus is occupied by } l\text{-th priority packet;} \end{cases}$   
 $\chi_2(t_m) = \begin{cases} 0, & \text{bus is empty;} \\ 1, & \text{bus occupation phase;} \\ 2, & \text{bus is occupied.} \end{cases}$
7. The set of continuous co-ordinates:  
 $z_v(t_m) = \{w(e''_1, t_m), w(e''_2, t_m)\}$ .
8. Initial state:  
 $w(e''_1, t_0) = \infty$ ,  $w(e''_2, t_0) = \infty$ ,  $\chi_1(t_0) < 0$ ,  $\chi_2(t_0) < 0$ .
9. Transition operators:  
 $H(e'_i), i = \overline{1, N}$ :  
 $\chi_1(t_{m+1}) = \begin{cases} x_i, & \chi_1(t_{m+1}) < x_i \wedge w(e''_1, t_{m+1}) \neq \infty \wedge w(e''_2, t_{m+1}) \neq \infty; \\ \chi_1(t_m), & \text{otherwise;} \end{cases}$   
 $\chi_2(t_{m+1}) = \begin{cases} t_m, & \chi_1(t_{m+1}) \leq 0; \\ \chi_2(t_m), & \text{otherwise;} \end{cases}$   
 $w(e''_1, t_{m+1}) = \begin{cases} t_m + \eta_m, & \chi_1(t_{m+1}) \leq 0; \\ w(e''_1, t_m), & \text{otherwise;} \end{cases}$   
 $w(e''_2, t_{m+1}) = \begin{cases} t_m + \xi_m, & \chi_1(t_{m+1}) \leq 0; \\ w(e''_2, t_m), & \text{otherwise.} \end{cases}$   
 $H(e''_1)$ :  
 $\chi_1(t_{m+1}) = 0$ ;  
 $\chi_2(t_{m+1}) = 0$ .  
 $G(e''_1)$ :  
 $Y = \{y_2\}, y_2 = \chi_1(t_m)$

$H(e_2'')$ :

$$\chi_2(t_{m+1}) = 2.$$

$G(e_2'')$ :

$$Y = \{y_1\}.$$

**Aggregate A<sub>5</sub>** (The aggregate describes an arbitration procedure in CAN)

1. The set of input signals:  
 $\{x_1\}$ , where  $x_1$  – a signal initiating an arbitration procedure, its value corresponds to a transferring message.
2. The set of output signals:  
 $\{y_1\}$ , where  $y_1$  – a signal indicating the end of an arbitration procedure, its value is a transferring message.
3. The set of external events:  
 $\{e_1'\}$ , where  $e_1'$  – the start of an arbitration procedure.
4. The set of internal events:  
 $\{e_1''\}$ , where  $e_1''$  – the end of an arbitration procedure.
5. Controlling sequence:  
 $e_1'' \rightarrow \{\eta_i\}_{i=1}^{\infty}$ , where  $\eta_i = \text{const}$ .
6. The set of discrete components:  
 $z(t_m) = \{\chi(t_m)\}$ , where  $\chi(t_m)$  – transmitted message.
7. The set of continuous co-ordinates:  
 $z_v(t_m) = \{w(e_1'', t_m)\}$ .
8. Initial state:  
 $w(e_1'', t_0) = \infty$ .
9. Transition operators:  
 $H(e_1')$ :  
 $\chi(t_{m+1}) = x_1$ ;  
 $w(e_1'', t_{m+1}) = t_m + \eta_m$ .  
 $G(e_1'')$ :  
 $Y = \{y_1\}, y_1 = \chi(t_m)$ .

**Aggregates A<sub>6</sub>, A<sub>7</sub>, A<sub>9</sub>** (The aggregate A<sub>6</sub> describes the transmission of informational part of a message and A<sub>7</sub> — an intermission between the transmission of messages; A<sub>9</sub> aggregate describes the transmission of control information of message)

The description of these aggregates is identical to the previous one with exception that the set of discrete co-ordinates of aggregate A<sub>7</sub> is empty.

The description of  $A_9$  aggregate is the same as of  $A_5$  excepting the set of output signal that is equal to  $\{y_1, \dots, y_N\}$  and transition operator  $G(e_1'')$ :  $Y = \{y_1, \dots, y_N\}, y_i = \chi(t_m), i = \overline{1, N}$ .

**Aggregate  $A_8$**  (The aggregate describes occupation of bus)

1. The set of input signals:  
 $\{x_1, x_2\}$ , where  $x_1$  – the start of a message transmission;  $x_2$  – the end of an intermission between message transmission.
2. The set of output signals:  
 $\{y_1, \dots, y_N\}$ , where  $y_i$  – a bus state.
3. The set of external events:  
 $\{e_1', e_2'\}$ , where  $e_1'$  – the start of a bus occupation,  $e_2'$  – the end of a bus occupation.
4. The set of internal events is empty.
5. Controlling sequence: none.
6. The set of discrete components:  
 $z(t_m) = \{\chi(t_m)\}$ , where  $\chi(t_m) = \begin{cases} 0, & \text{if a bus is idle;} \\ 1, & \text{otherwise.} \end{cases}$
7. The set of continuous co-ordinates:  
 $z_v(t_m) = \emptyset$ .
8. Initial state:  
 $\chi(t_0) = 0$ .
9. Transition operators:  
 $H(e_1')$ :  
 $\chi(t_{m+1}) = 1$ .  
 $G(e_1')$ :  
 $Y = \{y_1, \dots, y_N\}, y_i = \chi(t_m), i = \overline{1, N}$ .  
 $H(e_2')$ :  
 $\chi(t_{m+1}) = 0$ .  
 $G(e_2')$ :  
 $Y = \{y_1, \dots, y_N\}, y_i = \chi(t_m), i = \overline{1, N}$ .

**Aggregate Controller** (This aggregate describes the flow of messages, which have to be transmitted using CAN)

1. The set of input signals:  
 $\{x_1, x_2\}$ , where  $x_1$  – a bus state;  $x_2$  – a message is received.
2. The set of output signals:  
 $\{y_1\}$ , where  $y_1$  – a transmitted message, which value is its priority.

3. The set of external events:  
 $\{e'_1, e'_2\}$ , where  $e'_1$  – the change of a bus state,  $e'_2$  – next message was transmitted through a bus.
4. The set of internal events:  
 $\{e''_1\}$ , where  $e''_1$  – the end of a generation of a new message.
5. Controlling sequence:  
 $e''_1 \rightarrow \{\eta_i\}_{i=1}^{\infty}$ , where  $\eta_i$  – a duration between successful transmission of messages and the generation of a new message.
6. The set of discrete components:  
 $z(t_m) = \{\chi_1(t_m), \chi_2(t_m)\}$ , where  

$$\chi_1(t_m) = \begin{cases} 0, & \text{if a bus is idle;} \\ 1, & \text{otherwise;} \end{cases} \quad \chi_2(t_m) = \begin{cases} 0, & \text{if there are nothing to transmit;} \\ 1, & \text{otherwise.} \end{cases}$$
7. The set of continuous co-ordinates:  
 $z_v(t_m) = \{w(e''_1, t_m)\}$ .
8. Initial state:  
 $w(e''_1, t_0) = t_0$ ,  $a$  – the priority of transmitted messages from controller.
9. Transition operators:  
 $H(e'_1)$ :  

$$\chi_1(t_{m+1}) = x_1.$$
 $G(e'_1)$ :  

$$Y = \begin{cases} \{y_1\}, & \text{if } \chi_1(t_{m+1}) = 0 \wedge \chi_2(t_m) > 0; \\ \emptyset, & \text{otherwise;} \end{cases}$$

where  $y_1 = a$ .

 $H(e'_2)$ :  

$$\chi_2(t_{m+1}) = \begin{cases} \chi_2(t_m) - 1, & \text{if } x_2 = a, \\ \chi_2(t_m), & \text{otherwise;} \end{cases}$$

$$w(e''_1, t_{m+1}) = \begin{cases} t_m + \eta_m, & \text{if } \chi_1(t_m) = 0 \wedge x_2 = a, \\ w(e''_1, t_{m+1}), & \text{otherwise.} \end{cases}$$
 $H(e''_1)$ :  

$$\chi_2(t_{m+1}) = \chi_2(t_m) + 1.$$
 $G(e''_1)$ :  

$$Y = \begin{cases} \{y_1\}, & \text{if } \chi_1(t_{m+1}) = 0 \wedge \chi_2(t_m) = 1; \\ \emptyset, & \text{otherwise;} \end{cases}$$

where  $y_1 = a$ .

Presented above aggregate specification of CAN protocol was used to develop ESTELLE/Ag specification. The protocol was investigated using PRANAS-2 simulation system. The

relationship between an intensity of incoming flow of packets and their waiting time where investigated. Waiting time was calculated separately for the packets with different priorities.

### 3. Simulation results of CAN interface

Simulation results are presented in Figure 2, where *Waiting time* – the mean waiting time till packet transmission is started,  $\tau$  – mean time interval between two successfully of arrived packets to CAN interface. Each dependency in Figure 2 is for flow of packets with different priorities. The 2nd curve in Figure 2 is for the first priority flow (highest priority). The 1st curve is for the fourth priority flow (lowest priority).

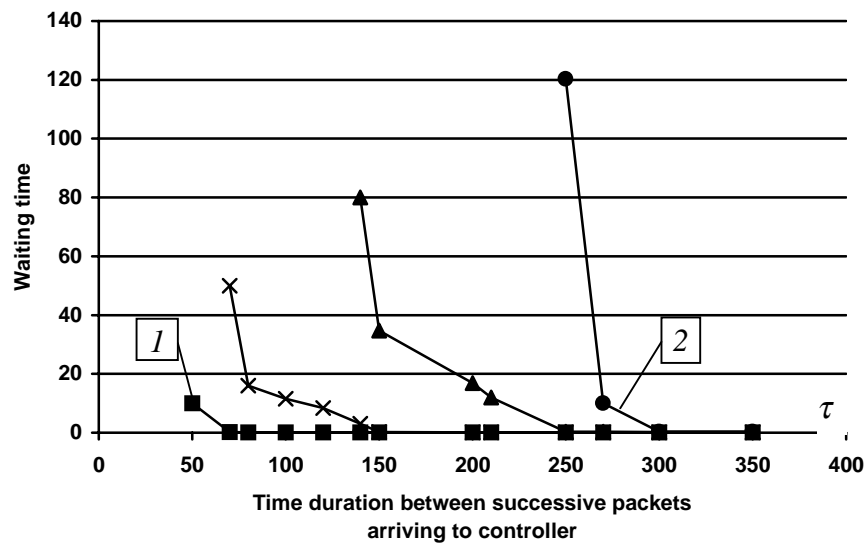


Figure 2. Waiting time dependencies (the scale of axis is measured in microsec)

Simulation experiments were carried out at the following parameters of CAN interface:

- Four flows of packets with different priorities and equal intensities were transmitted;
- Lengths of packets are the same and equal to 80 bits;
- Transmission rate of CAN monochannel is 1Mbps.

### 4. Requirements for packet transmission time through CAN interface

In order to prevent wheels locking, both response time of braking system mechanical part and transmission time of packets through CAN interface have to be less than time interval of wheel locking.

It is considered that braking system response time is approximately 0.06 sec. Figure 3 presents dependencies of car wheel angular velocity change during braking either on asphalt or an ice when car velocity is 72 km/h. According to Figure 3 [2] a wheel is locked after 0.07 sec on an ice while on asphalt locking lasts two-times longer.

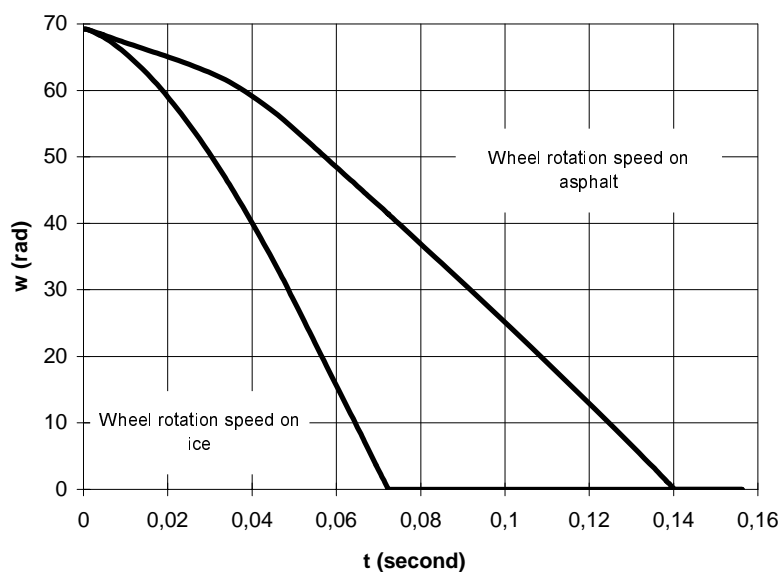


Figure 3. Dependencies of angular velocities of a car wheels

Considering wheel locking time at worst conditions (on an ice) we conclude that packets have to be transmitted not less than every 10 msec. Figure 2 shows that queues in CAN interface practically are not formed when time interval between successive packets arriving to controller is 10 msec. Duration of packet transmission in this case is 80  $\mu$  sec. This means that wheels braking process can be controlled without their locking if packets with information about each wheel state are transferred every 10 msec and transmission rate of CAN monochannel is 1Mbps.

## 5. References

- [1] ISO 11898:1993, Road vehicles - Interchange of digital information - Controller area network (CAN) for high-speed communication.
- [2] A. Mikuckas, D. Makackas. Modelling car braking temporal characteristics. Informacines technologijos'97, Technologija, Kaunas, 1997, pp. 218-224.
- [3] L.Motus, M.G.Rodd. Timing Analysis of Real-Time Software. Pergamon, 1994.
- [4] Philips Semiconductors. CAN Specification. Version 2.0, 1991.
- [5] H. Pranevicius, A. Chmieliauskas. Correctness analysis and performance prediction of protocol by means of aggregate approach and control sequences method. Academy of Science, USSR, Moscow, 1983, 32p.
- [6] H. Pranevicius, V. Pilkauskas, A. Chmieliauskas. Aggregate Approach for Specification and analysis of Computer network protocols. Technologija, Kaunas, 1994.
- [7] H. Pranevicius. Aggregate approach for specification, validation, simulation and implementation of computer network protocols. Lectures notes in Computer Sciences, No502, Springer-Verlag, Berlin, 1991, pp. 433-477.
- [8] SAE 81C90/81C91, Stand Alone Full CAN Controller. Siemens Semiconductor Group, 1994.
- [9] A. Valatka. Anti-lock braking and traction control systems of car. Smaltija, Kaunas, 1996.