ABCM Symposium Series in Mechatronics - Vol. 3 - pp.386-394

VERIFICATION AND VALIDATION OF AN ANALYSIS MODEL OF CAN-BASED NETWORKED CONTROL SYSTEMS

Eduardo Paciência Godoy, epgodoy@yahoo.com.br
Rafael Vieira de Sousa, rafael@cnpdia.embrapa.br
Arthur José Vieira Porto, ajvporto@sc.usp.br
Department of Mechanical Engineering, EESC - Engineering School of São Carlos, USP – University of São Paulo
Avenue Trabalhador São carlense, 400 CEP 13566-590, São Carlos, São Paulo, Brazil

Ricardo Yassushi Inamasu, ricardo@cnpdia.embrapa.br
EMBRAPA - Brazilian Agricultural Instrumentation Research Corporation
Street XV de Novembro, 1452 CEP 13560-970, São Carlos, São Paulo, Brazil

Abstract. *A major trend in industrial systems is to integrate computing, communication and control into different levels of manufacturing and information processes. One of the recent technology applied to perform these tasks is the networked systems with common bus architectures, called networked control systems (NCSs). A challeging problem in NCSs, such as CAN (Controller Area Network), is the network delay effects in the performance of the system. Network delays degrade the performance and destabilize the system. The performance parameters of NCSs that impact in the system include messages transmission time, response time and network utilization. These facts has motivated the development of timing analysis and models to calculate these delays and evaluate the performance of the NCSs. This work presents the verification and validation of an analysis model developed for CAN-based NCSs. The analysis model is composed by a CAN mathematical model and implemented in a simulation software. A real CAN network is used to support the data acquisition necessary for the verification and validation processes. These processes consist on the performance parameters comparison between the data obtained by the application of the simulation software and the data obtained using the CAN network. The experiments allows to evaluate the accuracy and the correct development of the proposed analysis model.*

Keywords: *mathematical model, networked control system, performance parameters, verification and validation*

1. INTRODUCTION

The traditional point-to-point communication architecture for control systems, which has been successfully implemented in industry for decades, is composed by a wire connecting the central control computer with each sensor or actuator point. However, a traditional centralized point-to-point control system is no longer suitable to meet new requirements, such as modularity, decentralization of control, integrated diagnostics, quick and easy maintenance and low cost. The introduction of common-bus network architectures can improve the efficiency, flexibility and reliability of these integrated applications through reduced wiring and distributed intelligence with consequent reducing in the installation, reconfiguration and maintenance.

Recent researches and developments have been released communication protocols for these networked systems with common bus architectures, called networked control systems (NCSs) (Yang, 2006). A preferred option is the Controller Area Network (CAN) (Bosch, 2006) which was originally developed to interconnect electronic unites in automotive area, but recently has also been considered in many other networked control applications (Othman et al., 2006). Nowadays, CAN-based networks are applied in distributed systems as the communication interface in proprietary architectures such as Device Net and CAN Open (CIA, 2006).

Despite the advantages and potentials, the existence of communication networks make the analysis and design of a NCS complicated. One of the fundamental challenges in this area is the development of analysis models and tools to predict the timing behavior and simulate the operation of the NCS (Lian, Moyne and Tilbury, 2002). Regardless of the type of network protocol used, the overall NCS performance is always affected by network delays (Lian, Moyne and Tilbury, 2002). The network induced delays and data packet dropouts occur when sensors, actuators, and controllers exchange data across the network. The characteristics of network-induced delays are mainly determined by the protocol used in the NCS. The performance metrics of networked systems that influence control requirements include messages transmission times, time delays and network utilization (Lian, Moyne and Tilbury, 2002).

Many works in recent years has been developing this analysis models and tools to calculate the delays and evaluate the performance of different types of protocols for NCSs (Torngren et al., 2006). In the case of CAN, the approach proposed in Tindell, Burns and Wellings (1995) became reference for this type of analysis. The importance of this kind of research and development for CAN-based NCS is demonstrated by recent works as de Frutos et al. (2006) and Cervin et al. (2006) that present the application of validated tools for analysis of NCSs.

Based on the cited facts and to supply the demand for these specific tools, this work presents the verification and the validation of an analysis model developed for CAN-based NCSs done in agreement to the diagram of the Fig. 1.

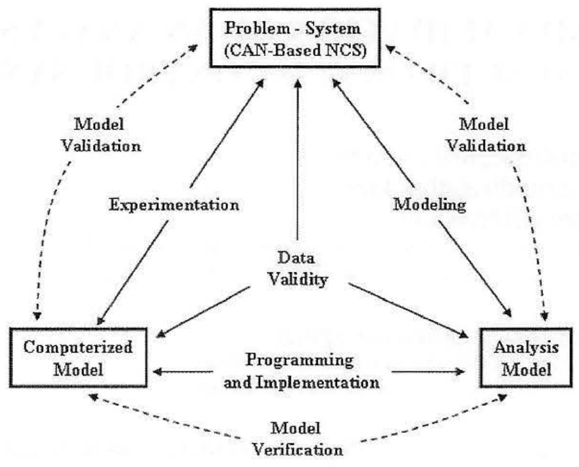


Figure 1. Diagram for the Development of an Analysis Tool (Modified from Sargent, 2005)

As described in Sargent (2005), model verification is often defined as "ensuring that the computer program of the computerized model and its implementation are correct" and model validation is usually defined to mean "substantiation that a computerized model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model. The analysis model is composed by a CAN mathematical model systemized in agreement to the equations firstly described in Tindell, Burns and Wellings (1995). The computer programming and implementation of the model was done with the LabVIEW environment resulting in a simulation software. A real CAN network is used to support the data acquisition necessary for the verification and validation processes. These processes consisted on the performance parameters comparison between the data obtained by the application of the simulation software and the data obtained using the real CAN network for an proposed experiment.

2. THE ANALYSIS MODEL

As described in Farsi, Ratcliff and Barbosa (1999), CAN is a serial communication protocol developed mainly for applications in the automotive industry but is also capable of offering good performance in other time-critical industrial applications. In a CAN-based network, data are transmitted and received using message frames that carry data from a transmitting node to one or more receiving nodes. An identifier, unique throughout the network, labels each message of the node and its value defines the priority of the message to access the network.

The CAN protocol is optimized for short messages and uses a CSMA/CD with NDBA (Carrier Sense Multiple Access / Collision Detection with Non-Destructive Bitwise Arbitration) arbitration access method. The bit stream of a transmission is synchronized on the start bit, and the arbitration is performed on the following message identifier, in which a logic zero is dominant over a logic one. The CAN protocol supports two message frame formats: standard CAN (version 2.0A, 11-bit identifier) and extended CAN (version 2.0B, 29-bit identifier) as shown in Fig. 2.

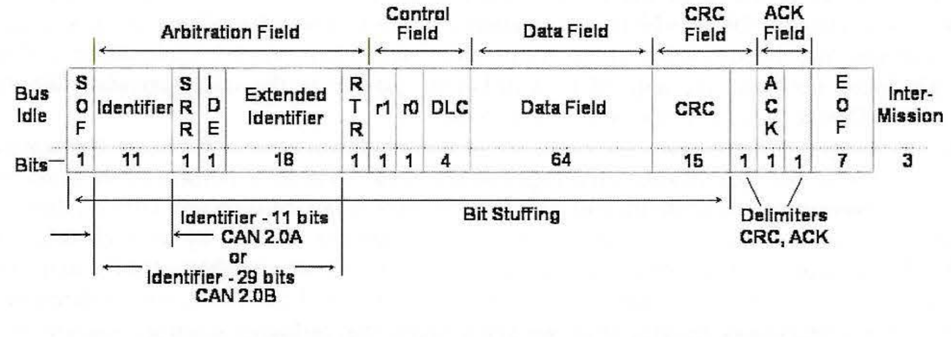


Figure 2. Message frame formats defined by the CAN protocol

2.1. Timing propeties of CAN networks

The time delays of a CAN-based network can be analyzed by studying their timing parameters (Lian, Moyne and Tilbury, 2002). Figure 3 shows a general timing diagram of the initialization and ending of the task of sending a CAN message over the network. The total time delay of a message, *Tdelay* , is defined as the difference between the time when the source node begins the process of sending a message, *Tsrc*, and the time when the destination node completes reception of this message, *Tdest* (i.e., *Tdelay* = *Tsrc* – *Tdest*).

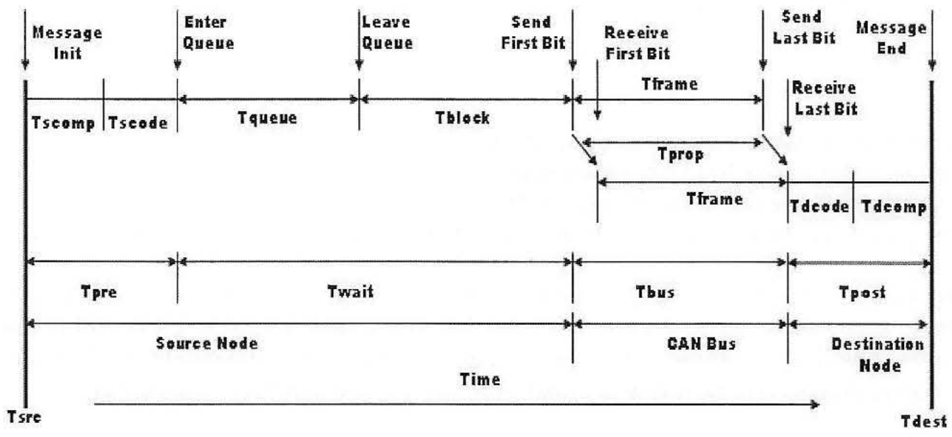


Figure 3. Timing diagram of the transmission of a CAN message (Modified from Lian, Moyne and Tilbury, 2001)

The total time delay, *Tdelay*, can be broken into three parts: time delays at the source node, *Tsrc*, on the CAN network, *Tbus*, and at the destination node, *Tdest*, as shown in Fig. 3. The *Tsrc* node includes the preprocessing time, *Tpre*, which is the sum of the computation time, *Tscomp* with the encoding time, *Tscode*. The waiting time, *Twait*, is the sum of the queue time, *Tqueue*, and the blocking time, *Tblock*. Depending on the amount of data that the source node must send and the traffic on the network, the waiting time may be significant. The queuing time, *Tqueue*, is the time that a message waits in the buffer at the source node while previous messages in the queue are sent. It depends on the blocking time of previous messages in queue, the periodicity of messages, and the processing load. The CAN network time delay, *Tbus*, includes the total transmission time of a message, *Tframe*, and the propagation delay of the network, *Tprop*. This will depend on message size, data rate, and the length of the network cable. The *Tdest* is the post processing time, *Tpost*, which is the sum of the decoding time, *Tdcode*, and the computation time, *Tdcomp*, at the destination node.

The *Tpre* and the *Tpost* times are typically constant and small. These times are totally related and dependent of the characteristics of the software and hardware used in the network. Because of these facts, a hypothesis of relation of them with the network *Jitter* (average waiting time of the message at the transmission queue before scheduling) was adopted. Finally, the Eq. (1) can express the total time delay that can be founded in the transmission of a message over the CAN network.

$$\begin{aligned} T_{delay} &= T_{src} - T_{dest} \\ &= T_{pre} + T_{wait} + T_{bus} + T_{post} \\ &= \underbrace{T_{pre} + T_{post}}_{J_m} + \underbrace{T_{wait}}_{Q_m} + \underbrace{T_{bus}}_{C_m} \end{aligned} \tag{1}$$

2.2. Modeling

The equations that define the parameters shown in Eq. (1) are described for CAN-based networks resulting in a relation between the Eq. (1) and the Eq. (2) (Godoy, 2007). From the timing analysis shown in more detail in Tindell, Burns and Wellings (1995), the Eq. (2) gives the worst-case transmission time and total delay of a message (*m*) in a CAN network.

$$R_m = J_m + Q_m + C_m \tag{2}$$

Where:

Jm: represents the message *Jitter* (empirically determined) with the value 0.1 ms (millisecond);
Qm: corresponds to the time spent by a message *(m)* in the waiting queue under error conditions;
Cm: represents the transmission time needed to physically transmit a message (*m*) over the CAN network.

The CAN data frame of a message (*m*) contains an amount of *O* bits of overhead and a bit-stuffing width of 5 bits. Only an amount of *T* of the *O* bits suffers stuffing (bit stuffing area shown in Fig. 1). The values of the variables *T* and *O* are different for the CAN message types. Godoy (2007) defines for CAN 2.0A, the values of 34 and 47 for *T* and *O* and for CAN2.0B, the values of 54 and 67 respectively. Finally, the Eq. (3) represents *Cm*.

$$C_m = \left(\left\lfloor \frac{T + 8S_m - 1}{4} \right\rfloor + O + 8S_m \right) . \tau_{bit} \qquad (3)$$

Where:
T: the amount of overhead bits subject to bit stuffing;
O: maximum number of overhead bits per message;
Sm = the size of a message *(m)* (8 bytes maximum);
τ bit = time spent to transmit one bit over the bus (founded with the CAN bus speed: bits per second).

The waiting time of a CAN message *(m)* in the transmission queue depends of the lower priority blocking time of the message that occupied the bus in the moment *(Bm)* and of the time needed to retransmit messages that presented error transmission (*Em*). It depends too of the higher priority messages transmission times. This time (*Qm*) is given by the recurrent relation presented in Eq. (4) with initial value $Q_m^0 = 0$ and iterations until convergence (i.e. $Q_m^{n+1} = Q_m^n$).

$$Q_m^{n+1} = B_m + \sum_{\forall j \in hp(m)} \left\lceil \frac{Q_m^n + J_j + \tau_{bit}}{T_j} \right\rceil . C_j + E_m(Q_m + C_m) \qquad (4)$$

Where:
hp (m): set of messages with higher priority than *(m)*;
Bm: the longest time that the given message *(m)* can be delayed by lower priority messages, given by the Eq.(5);
Tj: the sampling time of a given message *(j)*;
Jj: the *Jitter* of a message *(j)*.

$$B_m = \max_{\forall k \in lp(m)} (C_k) \qquad (5)$$

Where:
lp (m): set of lower priority messages than message *(m)* (if *m* is the lower priority message, *Bm* is equal to zero)
Cj, *Ck* are the same of *Cm* calculated by the Eq. (3).

In the Eq. (4) the term *Em(t)* is defined as a function of the error recovery and gives the superior limit of all overhead of error recovery that can occur in a time interval *(t)*. It is given by the Eq. (6) detailed in Punnekkat, Hansson and Norstrom (2000).

$$E_m(t) = \left(n_{error} + \left\lceil \frac{t}{T_{error}} \right\rceil - 1 \right) . (31 . \tau_{bit} + \max_{\forall K \in hp(m) \cup \{m\}} (C_k)) \qquad (6)$$

Where:
$n_{error:}$ number of errors that can occur in an arbitrary interval;
$T_{error:}$ the period of error occurrence.

The values of $n_{error} = 1$ and $T_{error} = 100$ms are defined as described in Tindell, Burns and Wellings (1995) for utilization in the Eq. (6). In every error that occurs in the network, the overhead of the error recovery can be increased in 31 bits, followed by a CAN message retransmission. Only messages with higher priority than message *(m)* can be retransmitted and cause a delay in message *(m)*. The biggest of those messages is given by the Eq. (7).

$$\max_{\forall K \in hp(m) \cup \{m\}} (C_k) \qquad (7)$$

Other parameter used to evaluate the performance of the CAN-based network is the network utilization rate (Lian, Moyne and Tilbury, 2002). The equation that defines this parameter is presented in the Eq. (8).

$$U = \sum_{i=1}^{N} \frac{Ci}{Ti} \qquad (8)$$

Where:
Ci: transmission time of the message (*i*) and given by (3);

Ti: sampling time of the message *(i)*;
N: the total number of messages in the CAN network.

The set of equations presented constitutes a mathematical model that can be used to analyze and evaluate performance of CAN–based NCSs.

2.3. Programming and Implementation

The mathematical model composed by the set of equations described was implemented in a computational program more detailed described in Godoy (2007). This implementation represents one useful task that facilitates the analysis of the performance data obtained with the utilization of the equations proposed. Figure 4 shows the utilization methodology for the simulation software development.

According to the proposed methodology, input data are defined (CAN configuration parameters) so that the equations that define the CAN mathematical model are used to obtain the output data (performance parameters) of the CAN-based network under analysis. Through the analysis of these output data it can be determined if the input parameters are correct defined and the operation of the CAN network is optimized or to guide a new choice of input data for other simulation.

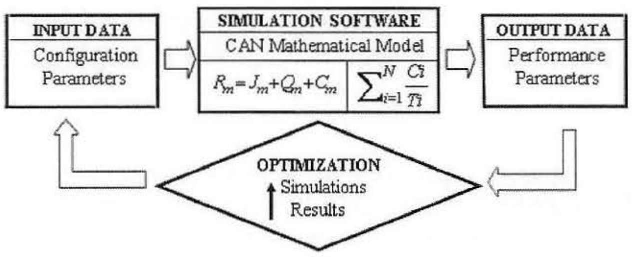


Figure 4. Utilization methodology for the simulation software of CAN networks

3. VERIFICATION AND VALIDATION PROCESSES

The development of the simulation software that implements the analysis methodology and the mathematical model was verified and validated in an experiment with a real CAN network. In this experiment is proposed the comparison between the results obtained with the application of the simulation software with the data acquired of a CAN-based network assembled in laboratory.

3.1. Assembly of the CAN-Based Network

For the proposed experiment, a CAN network composed by five ECUs (Electronic control unit) was assembled as shown in Fig. 5. The ECU used was described in Sousa (2002). Four ECUs (B, C, D, E) transmits data over the CAN bus for the analysis and one ECU (A) is connected to a notebook via RS232 (HMI – Human machine interface) for the data acquisition of the network. The RS232 interface was designed by configuring a specialized high-speed serial communication (57600 bps) in the microcontroller of the ECU. This ECU (A) had a single objective: to receive all messages from the CAN bus, calculate the network performance parameters and retransmit the results to the notebook.

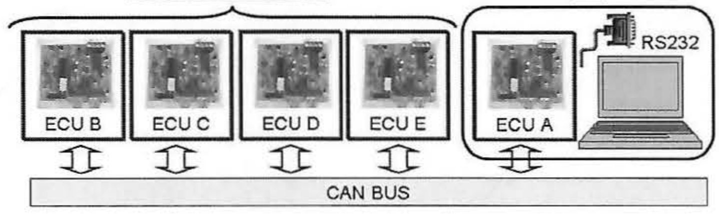


Figure 5. Schematic of CAN network for the experiment

3.2. Experimentation

Each ECU of the network analyzed, after its initialization, sends a specific periodic message over the CAN bus as shown in Table 1. The CAN 2.0B message format and the bus speed of 400 Kbits/s were selected for this experiment.

Table 1. Characterization of the CAN message data for the experiment

Nº	Message	Data Length - DL (Bits)	Priority	Sampling Time - ST (ms)
1	ECU B	16	1	50
2	ECU C	32	4	50
3	ECU D	48	3	50
4	ECU E	64	2	50

To perform a better comparison, other simulations for CAN network of the experiment were proposed. Table 2 specifies the parameters modified in relation to the original message data (Table 1) of the experiment network, for each simulation done.

Table 2. Parameters modified for the CAN message data of the experiment

Simulation	Parameters modified
S1	Original CAN message data (Table 1)
S2	Data length of the messages = 64 bits
S3	Priority scheme of the messages ([1,4,3,2] to [4,2,1,3])
S4	Messages sampling times (50ms to 100ms)

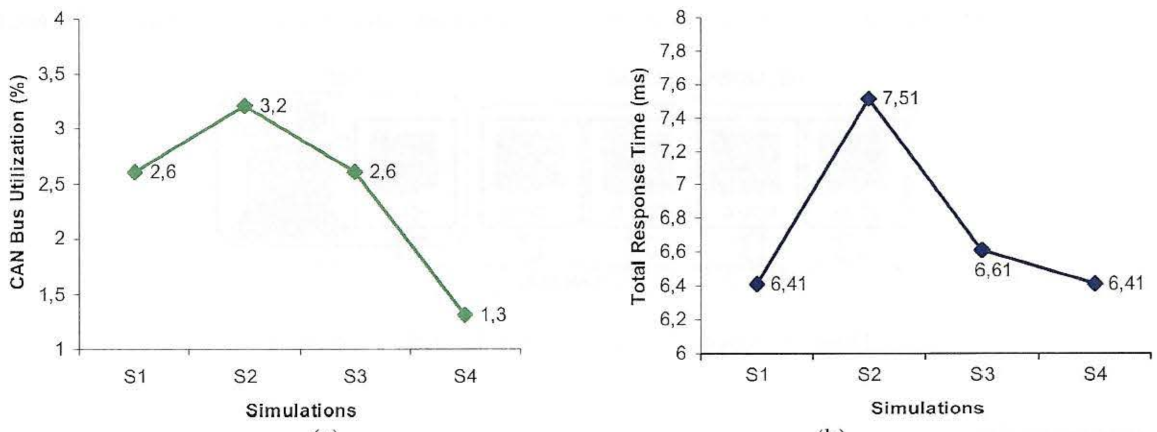
3.3 Data Validity

After the definition of the parameters for the simulations and following the steps of the experiment, the simulation software developed and the assembled CAN network were used with these input parameters to obtain the measurements. An example of the results obtained is shown in Table 3.

Table 3. Results of the application of the simulation software for the experiment – simulation S1

	Message Data			Message Transmission Times
Message	DL (Bits)	Priority	ST (ms)	Rm (ms)
ECU B	16	1	50	1,08
ECU E	64	2	50	1,58
ECU D	48	3	50	1,88
ECU C	32	4	50	1,88
Bus utilization	2,60 %	Total response time		6,41

Results as shown in Table 3 were obtained to the four simulations proposed in Table 2 with the simulation program and with the assembled CAN network. To simplify the presentation of these results, some graphs of the parameters total response time and bus utilization were traced. These graphs shown in Fig. 6, present the operational behavior of the CAN network of the experiment for the survey carried through with the data obtained with the application of the developed simulation software.



(a) (b)

Figure 6 – Operational behavior of the CAN network with results of the simulation software
(a) Graph of the bus utilization, (b) Graph of the total response time

The same simulation (S1, S2, S3, S4) are done with the CAN network assembled in the laboratory. Using the acquired data, some equivalent graphs to the Fig. 6 were traced, such as shown in Fig. 7.

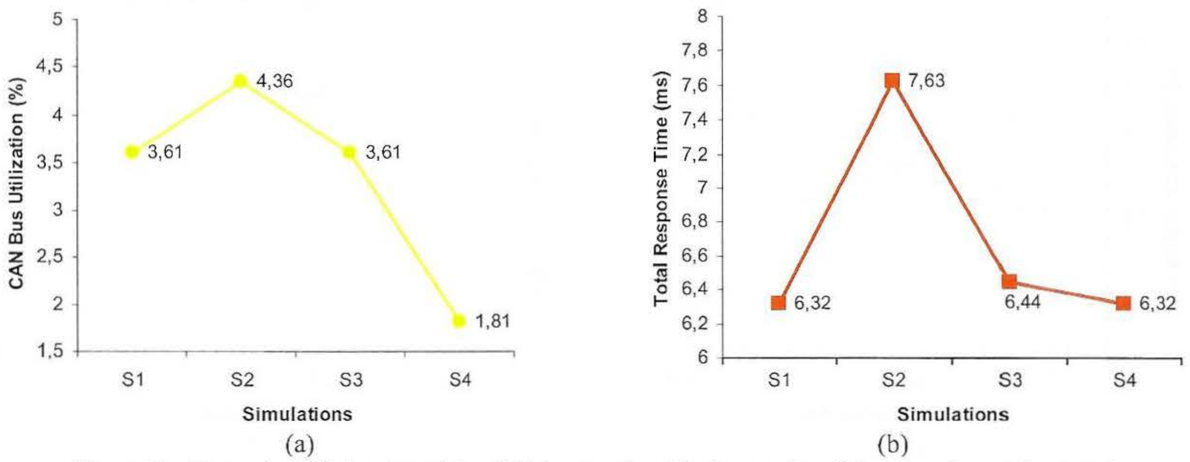


Figure 7 – Operational behavior of the CAN network with the results of the experimental network
(a) Graph of the bus utilization, (b) Graph of the total response time

The graphs shown in Fig. 8 and 9 present a summary of the results obtained for the four simulations proposed. These graphs were used to compare the results and behaviors of the CAN operation obtained for the both methods done (simulation software and CAN network assembled), verifying its similarities and proving and validating all the development done.

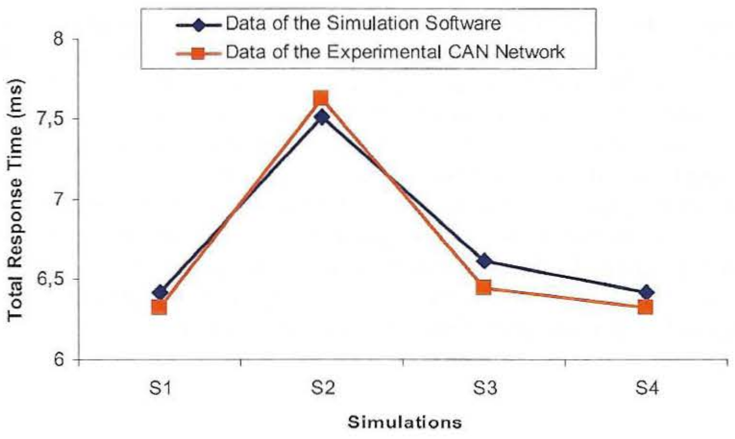


Figure 8 – Comparison of the results for the parameter total response time for the simulations of the CAN network

Analyzing the data of the Fig. 8, a small difference can be verified between the data obtained with the two methods for the parameter total response time. This fact can be explained because the mathematical model systemized considers the worst case for the operation of the CAN network. Thus, as the operation of the experimental CAN network not always occurs in accordance with the situation of worst case, the small difference between the values obtained was waited. Figure 8 presents a comparison between the results in the simulations for the parameter total response time. By the analysis of this results, can be verified that the data found for both methods have values very similar for the situations proposed in the simulations. Such results prove the correct development and validate the use of the implemented software.

Figure 9 represents a summary of the results obtained by the simulations for the parameter bus utilization. The difference founded between the values, showed in Fig. 9 (a), can be explained because the simulation software uses only the term C_m of the messages transmission times ($R_m = J_m + Q_m + C_m$) to calculate the parameter bus utilization (defined in Eq. (8)). However, with the assembled CAN network, because the impossibility to determine these values (J_m ,Q_m, C_m) separated, was used the term R_m (the time measured to transmit a message between two ECUs) to calculated the bus utilization. Thus, a bigger value was waited for the data obtained with the assembled CAN network in relation to the obtained with the simulation software. In Fig. 9 (b) an overlapping of the operational behaviors both methods is presented. This fact shows that although the differences found between the values, already explained, the curves of the behaviors are practically the same in both methods. In similar way to the previously conclusions, these facts prove the correct development and validate the use of the implemented software.

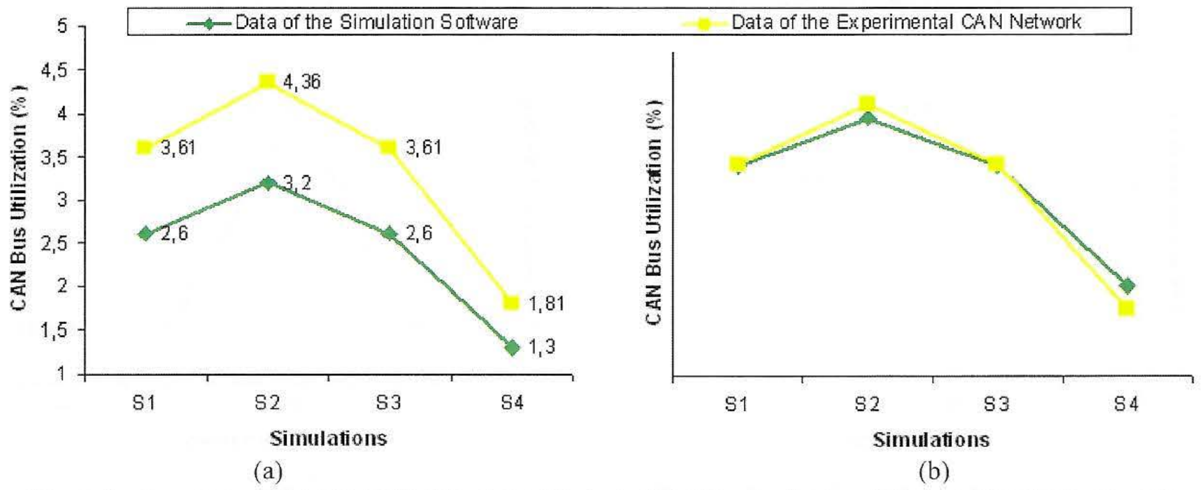


Figure 9 – Summary of the results for the parameter bus utilization for the simulations of the CAN network: (a) Comparison of the results, (b) Behaviors overlapped

The presented results demonstrate the verification and validation of the analysis model developed for CAN-based NCSs.

4. CONCLUSIONS

In this work was presented the verification and validation of an analysis model developed for CAN-based NCSs. The analysis model is based in a set of mathematical equations that provide information about the performance and operational behavior of the distributed system analyzed. The verification and validation processes consisted in the comparison between performance parameters obtained with the application of the simulation software with the obtained with the real CAN network assembled in laboratory. The results demonstrated the correct development of the analysis model and the software implementation of the model, determining that the model output behavior has sufficient accuracy for the model intended purpose over the domain of the model's intended applicability.

The detailed timing analysis and the analysis model presented to calculate the network parameters of CAN-based NCSs are the main contributions given by this article and should be useful for designers of NCSs. The possibility of use of the simulation software developed as a tool for performance analysis of CAN-based NCSs was proved in agreement to the correct programming and implementation of the analysis model systemized.

5. REFERENCES

Bosch, 2006, "CAN Specification Version", Available: http://www.can.bosch.com.

Cervin, A., Arzen, K.E., Henriksson, D., Lluesma, M., Balbastre, P., Ripoll, P. and Crespo, A., 2006, "Control Loop Timing Analysis Using True Time and Jitterbug", Proceedings of the 2006 IEEE Conference on Computer Aided Control Systems Design, Munich, Germany, 4-6 October 2006, pp. 1194-1199.

CIA, 2006, "Applications Fields of Controller Area Network", Available: http://www.cia-can.org.

de Frutos, J.A., Cadiñanos, E., Pérez, J.I. and García, S., 2006, "Tool for Analysis and Simulation of TTCAN Communication in Distributed Systems", Proceedings of the 30th Annual International Conference on Computer Software and Applications - COMPSAC'06, Vol. 2, pp. 119-122.

Farsi; M., Ratcliff, K. and Barbosa, M., 1999, "An overview of controller area network", Computing & Control Engineering Journal, Vol. 10, No. 3, pp.113-120.

Godoy, E.P., 2007, "Development of a Performance Analysis Tool of CAN-Based Networks for Application in Agricultural Systems (In Portuguese)", Msc. Dissertation, School Engineering of São Carlos, University de São Paulo, São Carlos, 93p.

Lian, F.L., Moyne, J.R. and Tilbury, 2002, "Network Design Consideration for Distributed Control Systems", IEEE Transactions on Control Systems Technology, Vol. 10(2), pp. 297-307.

Lian, F.L., Yook, J.K., Tilbury, D.M. and Moyne, J.R., 2006, "Network architecture and communication modules for guaranteeing acceptable control and communication performance for networked multi-agent systems", IEEE Transactions on Industrial Informatics, Vol. 2(1), pp. 12-24.

Lian, F.L.; Moyne, J.R. and Tilbury, D.M., 2001, "Performance evaluation of control networks: Ethernet, Controlnet and Device Net". IEEE Control Systems Magazine, p.66-83.

Othman, H.F., Aji, Y.R., Fakhreddin, F.T. and Al-Ali, A.R., 2006, "Controller Area Networks: Evolution and Applications", Proceedings of the 2nd International Conference on Information and Communication Technologies - ICTTA 06, Vol. 2, pp. 3088- 3093.

Punnekkat, S., Hansson, H. and Norstrom, C., 2000, "Response Time Analysis under Errors for CAN", Proceedings of the Real-Time Technology and Applications, Vol. 2, Washington, USA, pp. 258-266.

Sargent, R.G., 2005, "Verification and validation of simulation models", Proceedings of the 37th Winter Simulation Conference, Session: Advanced tutorials, Orlando, Florida, pp. 130- 143.

Sousa, R.V., 2007. "CAN (Controller Area Network): an approach to automation and control on agricultural area (In Portuguese)". Msc. Dissertation, Engineering School of São Carlos, University of São Paulo, São Carlos, 84p.

Tindell, K., Burns, A. and Wellings, A., 1995, "Calculating Controller Area Network (CAN) Message Response Time", Control Engineering Practice, Vol. 3, No. 8, pp. 1163-1169.

Torngren, M., Henriksson, D., Arzen, K.E., Cervin, A. and Hanzalek, Z., 2006, "Tool supporting the co-design of control systems and their real-time implementation: current status and future directions", Proceedings of the 2006 IEEE International Symposium on Intelligent Control, Munich, Germany, 4-6 October, pp. 1173-1180.

Yang, T.C., 2006, "Networked control system: a brief survey", IEEE Proceedings of Control Theory and Applications, Vol. 153, No 4, July, pp. 403 – 412.

6. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.