USING SIMULATION TOOLS IN THE DEVELOPMENT OF A NETWORKED CONTROL SYSTEMS RESEARCH PLATFORM

Eduardo Paciência Godoy, epgodoy@yahoo.com.br
Arthur José Vieira Porto, ajvporto@sc.usp.br
Department of Mechanical Engineering, EESC - Engineering School of São Carlos, USP – University of São Paulo
Avenida 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
Rua XV de Novembro, 1452 CEP 13560-970, São Carlos, São Paulo, Brazil

Abstract. *The introduction of fieldbus architectures in industrial systems can improve the efficiency and the reliability of the control system reducing time and costs of installation and maintenance. A current tendency in the research of fieldbus based control systems such as CAN (Controller Network Area) is the use of networked control systems (NCS). This new approach differs to the traditional fieldbus systems since the controller and the plant are physically separated and connected through an industrial network. The main challenges related to the development of NCS are the effects caused by the inclusion of the communication network in the closed loop control system. The presence of network delays between the sensors, actuators and controllers of the control system can degrade the performance and destabilize the system. To minimize these effects, simulation tools for NCS have been developed to assist the designer in the study of the network influence in the performance of the control system. These tools also allow the development and application of control methodologies to handle the network delay effect in these systems improving their performances and stability. This paper presents the application of two simulation tools, available in literature, in the development of a NCS research platform composed by several control systems connected through a CAN-based network. The tools are used to analyze the network delays effects, to obtain performance parameters and to design control methodologies for all control systems of the NCS platform. The results of the simulations allow the definition of design parameters such as sampling times and controller gains, and the verification of how sensitive the control loops are under various timing conditions including sampling times and network delays. Therefore, the utilization of simulations tools can ease the development of NCS and helps the designer to evaluate the control system prior to its implementation.*

Keywords*: networked control system, timing requirements, network delays, performance parameters*

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 networked control systems 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 (Moyne and Tilbury, 2007).

Recent applications of distributed fieldbus based control systems demonstrate a new approach for the use of industrial networks. In this type of application, called of Networked Control Systems (NCS), the controller, the sensor and the plant are physically separated and connected through a communication network (Yang, 2006) as shown in Fig. 1. The control signal is sent to the controller by a message transmitted over the network while the sensor samples the plant output and returns the information to the controller also transmitting a message over the network.

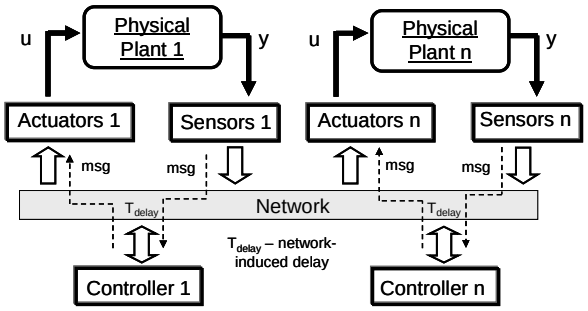


Figure 1. Structure of a NCS

Although the NCS offers several advantages over traditional centralized control systems, the existence of communication networks make the analysis and design of a NCS complex. Networked control systems impose additional problems inherent in control applications that are usually difficult to meet due to the variations and uncertainties introduced by the communication network: delays, *jitter*, bandwidth limitations and packet losses (Baillieul and Antsaklis, 2007). 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 (Lian et al., 2002). Godoy et al. (2008) describe the components of the network-induced delays in NCS and present the possible effects of the communication network in the performance and stability of NCS. Because of the network delays variability, the NCS can present characteristics of time variant systems doing the analysis and design of these systems even more complicated. Other problems related to the NCS are the correct choice of the network configuration parameters such as the network bandwidth, messages data length and messages sampling time. These parameters also influence the network performance (Lian et al., 2002). The NCS must sample and transmit data at a sampling time appropriate to achieve required performance metrics. However, if this sampling time is higher than the network bandwidth available, the network becomes overloaded, originating additional network delays and jitter, and causing packet losses and errors transmissions (Al-Hammouri et al., 2008). Therefore, NCS generally must meet two main criteria: bounded network delay and guaranteed transmission; that is, a message should be transmitted successfully within a bounded network delay. Unsuccessfully transmitted or large network delay in messages from a sensor to an actuator may deteriorate system performance or make systems unstable (Godoy et al., 2008).

Currently two main research directions can be distinguished in the NCS area. One focused in the development and use of specific tools to simulate the operation and to ease the analysis of the network influence in the performance and stability of the NCS (Cervin et al., 2003, Torngren et al, 2006; Al-Hammouri et al., 2008). And other in the development of control and design methodologies to handle the network effects improving the performance and guaranteeing the stability of the NCS (Zhang et al., 2001; Tipsuwan and Chow, 2003; Hespanha et al., 2007). Designing a NCS is essentially a multidisciplinary problem. Definitions made in the communication network design will affect the control design and vice versa. The use of these specific tools can allow verification and evaluation of how NCS problems affect the operation of the system and simulation of the timing behavior of the NCS prior to its implementation saving time and reducing design costs.

Following this guideline, this paper presents the application of two simulation tools, developed by the Lund University in Sweden, in the development of a NCS research platform composed by several control systems connected through a CAN-based network. The TrueTime and Jitterbug tools (Cervin et al., 2003), were used to analyze the network delays effects, to obtain performance parameters and to design control methodologies for all NCS of the platform. The results of the simulations allow the definition of design parameters such as message sampling times and controller gains and the verification of how sensitive the control loops are under various timing conditions including sampling times, network delays and jitter problems.

3. SIMULATION TOOLS FOR NCS

3.1 Background and Common Performance Metrics

Many works in recent years has been developing specific tools to help the analysis and design of NCS that merge knowledge of three topics: control systems, communication networks and real-time systems (Torngren et al., 2006; Al-Hammouri et al., 2008). The majority of these tools have been developed in the Matlab/Simulink environment, stimulated by its large acceptance and dissemination in the academic and industrial areas. Each one of these tools was developed in the academic area and is dedicated to one or a few tasks usually focused in one of these topics. The TrueTime and Jitterbug tools (Cervin et al., 2003) are specifically developed for control design tasks. In TrueTime the designer can model the entire NCS using a Simulink block library. This library allows the definition of parameters for the network protocol, the development of the control technique used and the selection of the plant model. The Jitterbug tool has a high level of abstraction in the definition of network delays and *jitter* but allow the calculation of a performance index related to the NCS quality of control. The AIDA (Redell et al., 2004) and TORSCHE (Sucha et al., 2006) are tools for design and analysis of real-time characteristics of NCS. The AIDA is used for timing analysis and the application of scheduling schemes for optimization of time response is the major purpose of the TORSCHE tool. The PiccSIM (Nethi et al., 2007) tools provide the functionality of use in the Internet. The designer can upload his control algorithm, define some parameters and use a few plant models for the simulation of the NCS. Despite of these Matlab based tools, it is not difficult to find tools for NCS developed in other platforms such as Labview (Pinnotti Jr and Brandão, 2005) and Modelica (Liu and Frey, 2008).

These specific tools provide facilities to analyze the behavior and operation of the NCS by means of simulation. The simulation eases the evaluation of results, because lets the replication of the experiment for the same model of the NCS. Varying only desired parameters in these replications, more specific information about the NCS can be achieved (Cervin et al., 2003). With this functionality, the designer can better evaluate the changes done or the defined

parameters in the NCS design according to common performance metrics used in these systems (Acton et al., 2006). Thus the evaluation of NCS has been usually focused in the following tasks:

- To analyze which factors (network delays, communication network parameters, signal sampling times) can affect the performance or stability of NCS;
- To analyze how factor is the most significant or that more influence in the design of NCS;
- To analyze performance metrics related to control and stability of the NCS such as settling time, rise time, overshoot, steady state error, stability margin and phase margin;
- To analyze performance metrics related to the communication network such as network utilization, network delays, network bandwidth and messages sampling times);
- To analyze performance metrics related to real-time systems such as *jitter*, processor utilization, response time, deadlines, latency, scheduling and prioritization.

3.2. TrueTime Tool

The TrueTime tool (http://www.control.lth.se/truetime) is a MATLAB/Simulink toolbox used to ease the simulation of operation and timing behavior of NCS, embedded systems or distributed systems with multiple controllers operating in real-time. The tool consists of a Simulink block library, basically with a computer kernel block and wired and wireless network blocks, as shown in Fig. 2, and a collection of Matlab MEX files. The entire NCS can be defined in the tool using the available blocks since the plant model, the controller algorithm until the communication network used (Cervin et al., 2003). The blocks are variable-step, discrete, MATLAB S-functions written in C++.

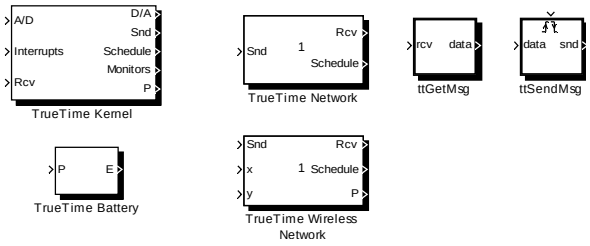


Figure 2. TrueTime Block Library

The kernel block simulates a real-time kernel executing user-defined tasks and interrupt handlers representing I/O tasks, A/D and D/A converters, control algorithms and network interfaces. The scheduling policy of the kernel block is arbitrary and decided by the user. The network blocks distribute messages between computer nodes according to a chosen network model. The blocks are connected with ordinary continuous-time Simulink blocks to form a real-time distributed communication system. All outputs are discrete-time signals (Ohlin et al., 2007).

The various network blocks allow nodes (kernel blocks) to communicate over simulated wired or wireless networks. The TrueTime network blocks simulate the physical layer and the medium-access layer of various local-area networks. The blocks only simulate the medium access (the scheduling), possible collisions or interference, and the point-to-point and broadcast transmissions. For wired networks, six models are supported: Ethernet switched Ethernet, CAN, token-ring, FDMA, and TDMA. The wireless network block supports simulation of the IEEE 802.11 WLAN and IEEE 802.15.4 ZigBee standards. TrueTime users have been developing high layer protocols using the network blocks for NCS simulations such as TCP based on the CSMA/CD and FlexRay based on the TDMA.

In the networks models implemented in TrueTime, only the interactions between nodes relevant for the timing behavior of the transmissions are modeled. That includes pre and post processing delays, collision detection and collision avoidance mechanisms, and probabilities of lost packets. A message contains information about the sending and the receiving computer node, arbitrary user data (typically measurement signals or control signals), the length of the message, and optional real-time attributes such as a priority (Ohlin et al., 2007).

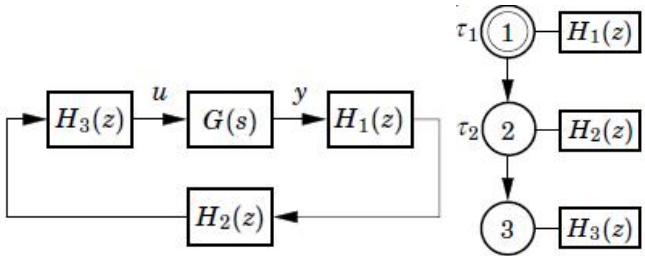
TrueTime simulation is programmed in the same way as a real NCS system. The application is written in Matlab code or in C++. The execution of tasks and interrupts are defined by code functions. Each task is defined by a set of attributes (priority, deadline, period, etc.) The main difference from real programming is that the execution/transmission times must be specified by the developer. This approach gives high flexibility to the TrueTime simulation tool.

3.3. Jitterbug Tool

The Jitterbug tool (http://www.control.lth.se/user/lincoln/jitterbug/) is a MATLAB toolbox that allows using the quadratic index defined by LQG theory to verify the network performance. The tool is composed by a set of Matlab functions that make interface to the *Control Systems* toolbox. These functions execute the tool initialization, configuration of NCS parameters and the calculation of a performance index or a cost function related to the control of the NCS. The cost function can easily be evaluated for a large set of design parameters and can be used as a basis for

the control and real-time design. Higher values of the cost function usually indicates more oscillatory or less stable closed loops of the NCS. Infinite cost means that the control loop is unstable. Jitterbug can be used to derive timing requirements from control performance specifications. The derived requirements are expressed in terms of the sampling times, latency, jitter, and network delays (Cervin and Lincoln, 2006). An important aspect in this tool is the possibility the compute a graphic of the spectral density of the timing conditions in the NCS related to the quality of control. For example, a curve relating the quality of control of a plant in closed loop to the sampling time and *jitter* of the NCS.

In the Jitterbug tool, the NCS is described by two models showed in Fig.3: the signal model and the timing model (Cervin and Lincoln, 2006). The signal model corresponds to the connection diagram of the continuous time and discrete time linear systems that compose the NCS (plant, controller, sensor and actuator). The timing model is quite simplistic and describes the execution sequence of the NCS discrete systems during a control period. The network delays in a control period are assumed to be independent from period to period. In the simplest case, a periodic timing model with random delays is used to describe the execution of the discrete-time systems, for example the control task. Figure 3 presents an example of the two models required for the use of the Jitterbug tool in a NCS. For NCS, the control system is described by four subsystems. The signal model showed in Fig 3(a) define the plant or process as a continuous time system (G), and the sensor (H1), the actuator (H3) and the controller (H2) as discrete time systems. Implicit in each discrete-time block is a sampler at the input and a zero-order-hold circuit at the output. The timing model showed in Fig. 3(b) define the execution of a control period of the NCS. The subsystem H1 or the sensor is executed first with a random network delay (τ1). After, the controller (H2) is executed with a defined network delay (τ2). Finally occur the execution of the subsystem H3 representing the actuator. The network delays in the timing model can be used to model delays induced by any kind of task in the NCS such as codification and processing times, scheduling times, and message transmission times in a communication network. This option provides a high flexibility of use for the tool because it can be applied to simulate any kind of network protocol for NCS. However, the network influence will be modeled in a simplistic way and the designer of the NCS or the tool user will have to know the better value distribution to model the network delays.



(a) (b)

Figure 3. Jitterbug Model of a NCS: (a) Signal Model (b) Timing Model

4. DEVELOPMENT OF THE NCS PLATFORM

4.1. Proposed NCS Platform

According to Pinotti Jr and Brandão (2005), the increasing use of distributed control systems with common bus architectures in industrial control has demanded the inclusion of fieldbus systems in academic teaching and educational laboratories. Many works have been developing this platforms for fieldbus research and teaching (Martín and Tadeo, 2006; Kolla, 2007). But for NCS, that represents a different application of fieldbus technologies, these platforms are not so frequent. The platform proposed in this paper is being developed for NCS study and would represent a small-scale manufacturing system composed by several closed loop control systems. The communication network used for the control systems integration and information exchange is the CAN protocol. The architecture of the proposed NCS platform is shown in Fig. 4. Common control systems used in the industrial area such as DC motor velocity and position control, temperature control, tank level control and belt conveyor control are selected for the platform. Each of the defined systems has an electronic control unit (ECU) responsible for the data acquisition, actuation in the plant and communication with the CAN-based network. Two desktops with LabVIEW and PCI-CAN interfaces from National Instruments are used for the development of the control methodologies of the NCS. The architecture proposed has high flexibility for the research and teaching of NCS. In addition to the aimed tasks for NCS such as analysis, modeling, simulation and control, the platform provides capabilities to study and applies advanced techniques for NCS development. Among these techniques are the real-time evaluation, hardware-in-the-loop simulation and rapid control prototyping.

For each of the NCS that composes the platform, the time-driven sensor node samples the plant or process periodically and sends the samples to the controller node over the CAN network. Upon receiving a sample, the controller computes a control signal which is sent to the actuator node, where it is subsequently actuated. The threads executing in the controller and actuator nodes are both event-driven. All the closed loop control systems in the platform

are sharing both limited CAN-based network bandwidth and controller CPU. The competition for these constrained resources will certainly increase the network-induced delays of the control loops can degrading the overall performance of NCS in the platform.

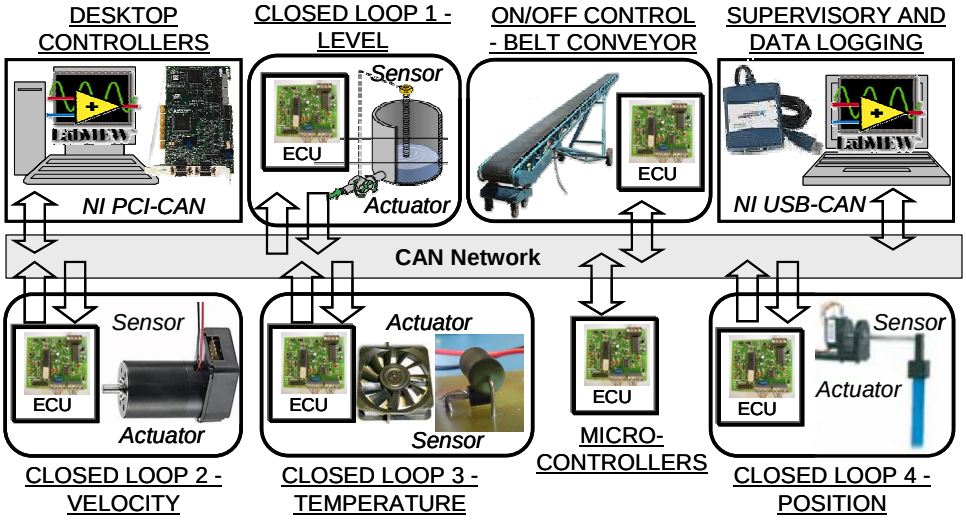


Figure 4. Architecture of the NCS Platform

The chosen of the CAN protocol (Bosch, 2006) for the communication network due to its low cost of development and large acceptance in the industrial and academic areas. The CAN was originally developed to interconnect electronic control units (ECU) in automotive area, but recently has also been applied in many other networked control applications. Currently, CAN-based networks are applied as a solution for distributed control systems in several areas, such as robotic, automated manufacturing and process control environments, and used in proprietary architectures such as Device Net and CAN Open (CIA, 2006). As described in Johansson et al., (2005), in CAN-based networks 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.

4.2. Simulation and Analysis

The use of simulation tools is necessary for the development of the NCS platform because they help the designer to evaluate the control system prior to its implementation saving time and reducing design costs. Selecting a tool for NCS design depends on the system specifications and the required results. In consequence, the TrueTime and Jitterbug tools were selected to be used in the NCS platform development. For the platform proposed, the TrueTime tool are used to analyze the CAN network delays effects, to obtain performance parameters and to design control methodologies for all control systems. The Jitterbug tool is used to evaluate how sensitive one control loop of the platform is under various timing conditions imposed by the dynamics of the whole NCS platform operation.

Firstly the entire model of the NCS platform was developed in the TrueTime tool. The mathematical model of each control system was obtained by available equations and experimental procedures. In a known method, a step response graph of the open loop plants was used too for the parameters estimation of the plants transfer functions. The belt conveyor system was not included in the platform model because it uses an on/off controller. Its influence in the platform operation was included by a node that only sends and receive messages in the CAN network. The DC motor used for velocity control was a Motron M-910 (24V). The Eq. (1) to (4) are the basic relations for mathematical modeling of DC motors.

$$V = R.I + L.\frac{dI}{dt} + V_{emf} \tag{1}$$

$$V_{emf} = K_e.w_m \text{, with } w_m = \frac{dq_m}{dt} \tag{2}$$

$$T_m = h_m.K_m.I \tag{3}$$

$$T_m = J_m.\frac{dw_m}{dt} + B_{eq}.w_m \quad (4)$$

With the parameters: R = motor resistance (Ω), V = voltage applied to the motor (V), I = current applied to the motor (A), L = motor inductance (H), V_{emf} = back emf voltage (V), Ke= motor emf constant (V.s/rad), w_m = angular velocity of the motor shaft (rad/s), θ_m = angular rotation of the motor shaft (rad), K_m = motor torque constant (N.m/A), T_m = torque applied to the motor shaft (N.m), η_m = motor efficiency (%), J_m = inertia of the motor shaft (kg.m^2), B_{eq} = equivalent viscous friction (N.m.s/rad).

Applying the Laplace transform and rearranging the equations, the transfer function for the DC motor velocity control can be done by the Eq. (5). The final transfer function was obtained with the replacement of the parameters from datasheets and experimental procedures and disregarding the effects of the lower values of the motor inductance.

$$\frac{w_m}{V}(s) = \frac{h_m.K_m}{J_{eq}.L.s^2 + (B_{eq}.L + J_{eq}.R).s + K_m.K_e + B_{eq}.R} = \frac{0{,}00886}{0{,}007982.s + 0{,}008144} \quad (5)$$

For the position control, a DC motor Maxon REmax (24V) with planetary gearbox was selected to be used. The mathematical modeling for this case needs additional relations. For this purpose, the Eq. (6) to (9) was used with the basic relations described earlier in Eq. (1) to (4).

$$q_m = k_r.q_r \quad (6)$$

$$J_m.\frac{d^2q_m}{dt} = T_m + \frac{T_r}{h_r.k_r} \quad (7)$$

$$J_r.\frac{d^2q_r}{dt} = T_r - B_{eq}.\frac{dq_r}{dt} \quad (8)$$

$$J_{eq} = J_r + h_r.k_r^{\ 2}.J_m \quad (9)$$

With the parameters: k_r = gearbox reduction (109:1), θ_r = angular rotation of the gearbox shaft (rad), T_r = torque in the gearbox output (N.m), η_r = gearbox efficiency (%), J_r = inertia of the gearbox (kg.m^2), J_{eq} = equivalent inertia of the system (kg.m^2).

Using the equations and applying the Laplace transform, the transfer function for position control of a DC motor with gearbox is showed in the Eq. (10). The transfer function was obtained with the replacement of the parameters from datasheets and experimental procedures and disregarding the effects of the lower values of the motor inductance.

$$\frac{q_r}{V}(s) = \frac{h_r.k_r.h_m.K_m}{J_{eq}.L.s^3 + (B_{eq}.L + J_{eq}.R).s^2 + (h_r.k_r^{\ 2}.h_m.K_m.K_e + B_{eq}.R).s} = \frac{1{,}318}{0{,}02476.s^2 + 3{,}509.s} \quad (10)$$

The mathematical modeling for the one reservoir level control uses the relations given by the mass balance and turbulent flow in systems. Thus, the Eq. (11) and (12) were used for the modeling. The linearization of the nonlinear Eq. (11) can be done in a selected operation point ($\left(\bar{h}, \bar{q}\right)$, in Eq. (13), considering that errors in the model for low variations around this point are insignificant, resulting in the final Eq. (14).

$$A.\frac{dh}{dt} = q_i - q_o \quad (11)$$

$$q_o = k.\sqrt{h} \quad (12)$$

$$R = \frac{dh}{dq} = \frac{2.\bar{h}}{\bar{q}} = \frac{2.\sqrt{\bar{h}}}{k} \quad (13)$$

$$A.\frac{dh}{dt} = q_i - \frac{2}{R}.h \quad (14)$$

With the parameters: q_i = input flow of pump in reservoir [cm^3/s], q_o = output flow of reservoir [cm^3/s], h = level in reservoir [cm], k = valve characteristics considering turbulent flow (obtained experimentally k = 8), A = cross section of the reservoir (179,5 cm^2).

Applying the Laplace transform and rearranging the equations, the transfer function for the reservoir level control can be done by the Eq. (15). The final transfer function was obtained with the replacement of the parameters from experimental procedures and using the selecting point $\bar{h}$ =9 cm.

$$\frac{h(s)}{q_i(s)} = \frac{1}{A.s + \frac{2}{R}} = \frac{R/2}{A.R.s/2 + 1} = \frac{0{,}75}{67{,}31.s + 1} \tag{15}$$

The mathematical modeling for the temperature control uses the relations given by the energy conservation and convection heat transfer in thermal systems. Thus, the Eq. (16) to (18) were used for the modeling.

$$q_i - q_o = C.\frac{d(T_s - T_a)}{dt} \tag{16}$$

$$q_o = \frac{1}{R}.(T_s - T_a) \tag{17}$$

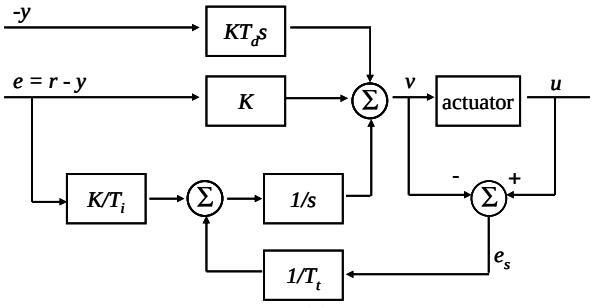
$$T = T_s - T_a \tag{18}$$

With the parameters: q_o = output heat flow [W], q_i = input heat flow (using a 20W and 3,9 ohms resistor) [W], T_s = temperature measured by the sensor [°C], T_a = ambient temperature [°C], T = temperature change [°C], R = thermal resistance [°C/W], θ = dead time related to the step response of the system [s], C = thermal capacitance (J/°C].

Using the equations described plus the common dead time inherent to thermal systems and applying the Laplace transform, the transfer function for temperature control can be done by the Eq. (19). The final transfer function was obtained with experimental procedures based on the step response graph.

$$\frac{T(s)}{q_i(s)} = \frac{R}{R.C.s + 1}.e^{-q.S} = \frac{17{,}19}{78{,}42.s + 1}.e^{-S} \tag{19}$$

With the plant models for the NCS of the platform, the controllers could be designed for each system. These controllers have to consider all issues presented in the NCS such as actuators saturation and sampling times compatible to the hardware's used. According to Pohjola (2006), controllers for NCS cannot be designed with classical control theory of continuous time systems because the resulting performance is unsatisfactory. The controllers for NCS have to handle the network delay effects in the systems. Based on the flexibility and large application, a PID controller was defined to be designed for the systems in the NCS platform. The controller is a discrete-time PID controller derived with the backward derivative approximation and with setpoint weighting, reference off and filtering on derivative part. This controller has been applied for NCS with good results (Pohjola, 2006). In this work, an anti-windup of the integral action was added to this controller to work with saturation of the actuators where the parameter T_t is equal to $\sqrt{T_i.T_d}$ for PID controllers and equal to T_i for PI controllers. Thus, the resulting discrete-time controller with constant sampling time (h) and constant filtering of the derivative part (N) is showed in the Fig. 5:



(a)

$$e(k) = r(k) - y(k)$$

$$u_p(k) = K.e(k)$$

$$u_i(k) = u_i(k-1) + \frac{K.h}{T_i}e(k) + \frac{K.h}{Tt}e_s(k-1)$$

$$u_d(k) = \frac{T_d}{T_d + Nh}u_d(k-1) + \frac{KT_dN}{T_d + Nh}\left[y(k) - y(k-1)\right]$$

$$u(k) = u_p(k) + u_i(k) + u_d(k)$$

(b)

Figure 5. Discrete-Time PID Controller for NCS: (a) Schematic (b) Parameters Implementation

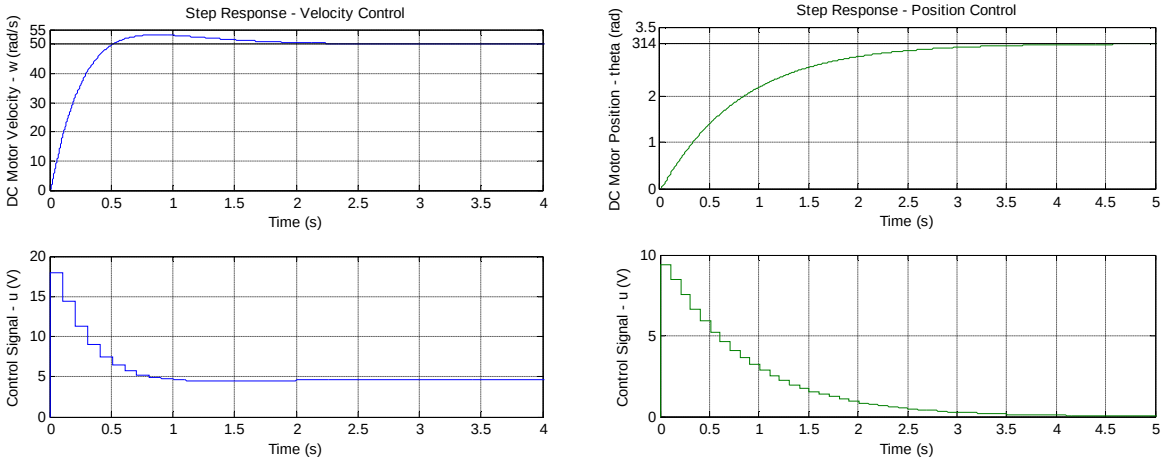
With all systems (controllers, actuators and sensors) of the platform defined and implemented in the TrueTime tool, several simulations of the NCS platform operation were done. A network transmission speed of 250kbit/s and messages data length of two bytes for all the ECUs were defined for the CAN network. Dealing to the hardware used in the ECUs (PIC18F258 microcontrollers), a sampling time of 100ms (0.1s) was determined for the NCS of the platform. A value of

N=10 was used in the constant filtering of the controllers. The objective of the simulations were to design and tuning the control methodologies based on PID controllers and to obtain performance parameters such as overshot and settling time (5% criteria) that achieves the requirements defined for all control systems of the NCS platform. For example, the main requirement for the NCS of DC motor position control is do not present overshoot. After some simulations and analysis, the performance achieved for the NCS of the platform are showed in Fig. 6 and 7.

The Tab. 1 presents a synthesis of the information obtained after the simulations of the NCS platform. The best performance metrics achieved for all of the NCS are showed. The PID controller gains could have been defined in agreement to other necessary design parameters such as sampling time. Their values are also showed in the Tab 1.

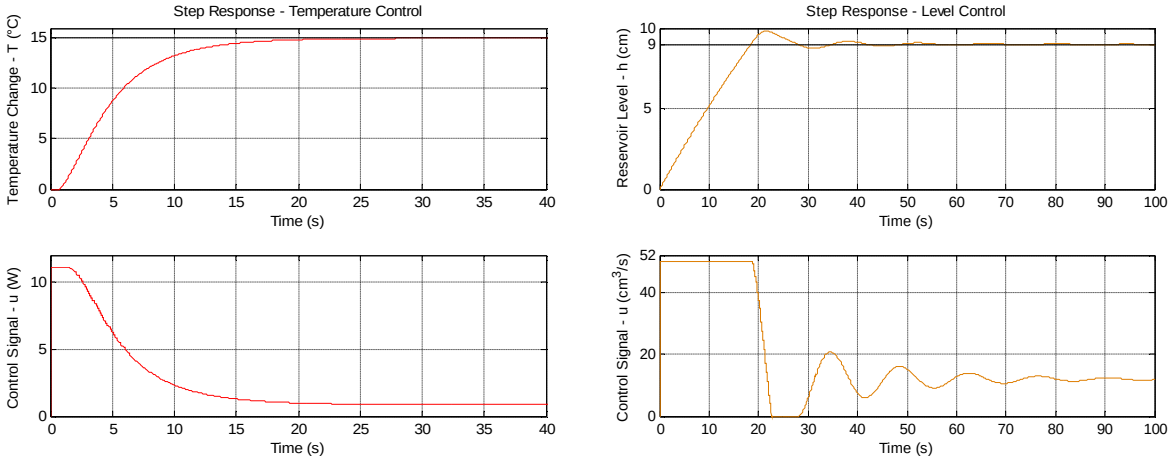
Table 1. Information about PID Controllers and Performance for the NCS of the Platform

NCS of the Platform	Controller Parameters			NCS Performance Measures			
	K	Ti	Td	Overshoot (%)	Peak time (s)	Rise Time (s)	Settling Time (s)
DC Motor Velocity Control	0,36	0,65	0	6,5	0,91	0,52	1,52
DC Motor Position Control	3	0	0,001	-	-	1,9	2,9
Temperature Control	2	7	0,01	-	-	10,7	13,4
Level Control	10	0,55	0	9,03	21,71	18,48	33,1



(a) (b)

Figure 6. Performance of the DC Motors NCS: (a) Response and Control Signal for Velocity Control (b) Response and Control Signal for Position Control



(a) (b)

Figure 7. Performance of the other NCS: (a) Response and Control Signal for Temperature Control (b) Response and Control Signal for Level Control

The simulations allow verify that several parameters can influence the performance of a NCS. Among these parameters are the network speed and load, message prioritization, network delays and sampling time. But the most important parameter that affected the NCS of the platform was the sampling time. To evaluate the control sensitivity of the NCS under different sampling times the Jitterbug was used. This tool calculates a performance index related to the

control of the NCS. Higher values of the cost function usually indicates more oscillatory or less stable closed loops of the NCS or even more its instability. The network delays in the NCS of the platform were modeled by the two random variables τ_1 and τ_2. The total delay from sampling to actuation is given by $\tau_{total} = \tau_1 + \tau_2$. As a cost function, the sum of the squared process input and the squared process output was chosen determining the Eq. (20).

$$J = \lim_{T\to\infty} \frac{1}{T} \int_0^T [y^2(t) + u^2(t)]dt \tag{20}$$

Figure 8(a) and (b) show the results for two (the DC motor position control and the level control) of the NCS of the platform. The graph in these figures presents the values of the cost function or the quality of control of the NCS related to different timing conditions of sampling times and network delays. The analysis of these graphs allows verify that despite the sampling time is the most significant factor in the design of NCS, its influence was related to the kind of system. In Fig. 8(a) the NCS for DC motor position control represents a NCS with fast dynamics. In this type of system, the influence of the sampling time is big. Looking in the graph, small variations in the sampling time can hardly affect the performance of the system. The NCS was simulated for sampling times from 0 to 0,5s. This conclusion could be verified too in simulations for the DC motor velocity control, which represents a system with fast dynamics too.

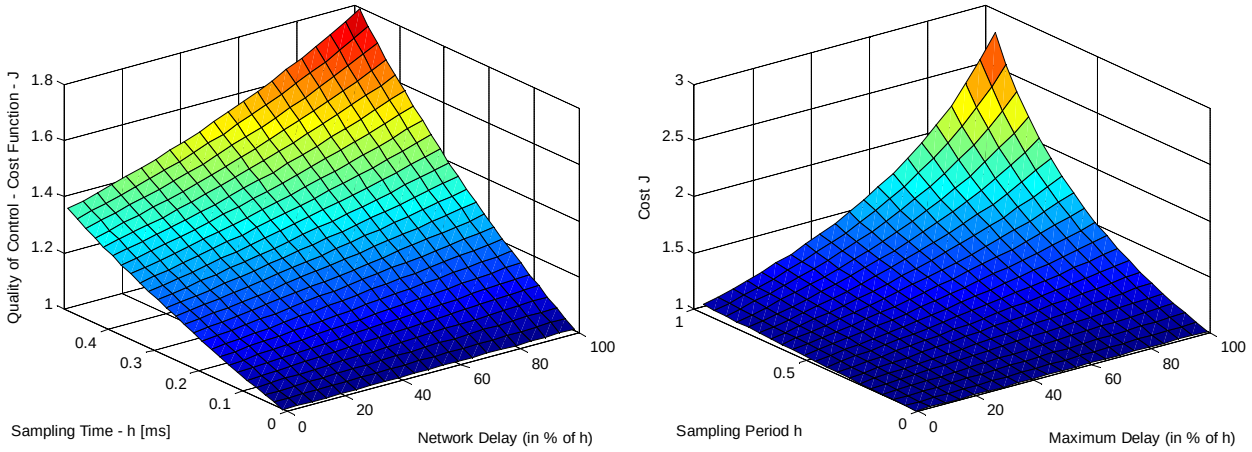


Figure 8 Sensitivity of two NCS of the Platform under Different Timing Conditions: (a) Sensitivity of the NCS Position Control (b) Sensitivity of the NCS Level Control

On the other hand, the influence of the sampling time in NCS with lower dynamics is smaller. In the graph of the Fig. 8(b) that represents the results of the NCS for level control, this fact can be verified. The effect in the performance of this type of NCS is smaller for bigger variations in the sampling time. Look that the NCS was simulated for sampling time from 0 to 1s.

Future work will be done to finish the construction and integration of all NCS of the platform. With the platform working in laboratory, the designed controllers will be implemented in each NCS and several experiments could have been made to verify the controller design and evaluate the use of simulation tools for NCS.

5. CONCLUSIONS

Designing a NCS is essentially a multidisciplinary problem. Definitions made in the communication network design will affect the control design and vice versa. Because of this, the use of simulation tools is necessary for the development of the NCS platform. In this paper was presented the application of two simulation tools for the development of a NCS research platform. A brief revision about specific tools developed to help the design of NCS and common performance metrics used to evaluate its performance were also presented. The TrueTime tool was used to analyze the network delays effects and to evaluate the influence of design parameters such as sampling times in the performance of the NCS. The application of this tool also allows the design of control methodologies for all NCS of the platform. The proposed PID controllers were designed and the controller gains are determined in agreement to necessary design requirements such as sampling time, overshoot and settling time.

The results of the simulations in Jitterbug allow to evidence that the sampling time is one of the major parameters that affect NCS performance and to verify the sensitivity of the NCS under various timing conditions including different sampling times and network delays. An important conclusion is that the influence of the sampling time is bigger in NCS with fast dynamics (for example DC motor control) than in NCS with low dynamics (for example level control). As a

result, the utilization of the simulations tools facilitated the development of NCS platform and helped the designer in the evaluation of the NCS prior to its implementation saving time and reducing design costs.

6. ACKNOWLEDGEMENTS

The authors acknowledge the FAPESP - The State of São Paulo Research for the support to this paper.

7. REFERENCES

Acton, K., Antolovic, M., Kalappa, N., Luntz, J., Moyne, J. and Tilbury, D., 2006, "Practical Metrics for Evaluating Network System Performance", Proceedings of the Network Performance Workshop, University of Michigan.

Al-Hammouri, A., Branicky, M.S and Liberatore, V., 2008, "Co-simulation Tools for Networked Control Systems", Springer-Verlag, pp. 16-29, Berlin.

Baillieul, J. and Antsaklis, P.J., 2007, "Control and Communication Challenges in Networked Real Time Systems", Proceedings of IEEE Technology of Networked Control Systems, Vol. 95, No. 1, pp. 09-28.

Bosch, 2006, "CAN Specification Version 2.0", Disponível em: <http://www.can.bosch.com>, Acesso em: Julho, 2006.

Cervin, A. and Lincoln, B. 2006, "Jitterbug 1.21 Reference Manual". Dept. of Automatic Control, Lund University.

Cervin, A, Henriksson, Lincoln, Eker, J. and Arzen, K., 2003, "How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime", IEEE Control System Magazine, Vol. 23, pp. 16-30.

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

Godoy, E.P, Porto, A.J.V and Inamasu, R.Y. 2008, "Ferramentas de Analise, Simulação e Projeto de Sistemas de Controle via Redes: Uma Revisão", Anais do V Congresso nacional de Engenharia Mecânica CONEN, Bahia.

Hespanha, J.P., Naghshtabrizi, P. and XU, Y., 2007, "A Survey of Recent Results in Networked Control Systems", IEEE Proceedings of the Technology of Networked Control Systems, Vol. 95, No. 1, pp. 138-162.

Johansson, K.H., Torngren, M. and Nielsen, L., 2005, "Vehicle applications of controller area network", Handbook of Networked and Embedded Control Systems, Ed. Birkhäuser, 25p.

Kolla, S., 2007,"CAN-based Fieldbus Experiments", Proceedings of the 2007 ASEE Annual Conference & Exposition, American Society of Enginnering Education, Honolulu, Hawai, 20-24 June.

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

Liu, L. and Frey, G., 2008, "Feasibility Analysis for Networked Control Systems by Simulation in Modelica", Proceedings of the 13th International Conference on Emerging Technologies and Factory Automation, pp. 729-732.

Martín, J. V.and F. Tadeo, 2006, "Learning fieldbus technology using open and flexible educational equipment", Proceedings of the Seventh IFAC Symposium on Advances in Control Education, Vol. 7(1).

Moyne, J.R. and Tilbury, D.M., 2007, "The Emergence of Industrial Control Networks for Manufacturing Control, Diagnostics, and Safety Data". IEEE Technology of Networked Control Systems, Vol. 95, No. 1, pp. 29-47.

Nethi, S., Pohjola, M., Eriksson, L. and Jantti, R., 2007, "Platform for Emulating Networked Control Systems in Laboratory Environments", Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, Helsinki, Finland, June 18-21.

Ohlin, M.; Henriksson, D. and Cervin, A. (2007). TrueTime 1.5 Reference Manual. Internal Report, Department of Automatic Control, Lund University, Sweden, January 2007.

Pinotti Jr, M. and Brandão, D., 2005, "A Flexible Fieldbus Simulation Platform for Distributed Control Systems Laboratory Courses", International Journal of Engineering Education, Vol. 21, No. 6, pp. 1050-1058.

Pohjola, M. (2006). PID Controller Design in Networked Control Systems, Master Dissertation, Department of Automation and Systems Technology, Helsinki University of Technology, 91p.

Redell, O., El-Khoury, J. and Torngren, M., 2004, "The AIDA tool-set for design and implementation analysis of distributed real-time control systems", Journal of Microprocessors and Microsystems, Vol. 28:4, pp. 163-182.

Sucha, P., Kutil, M., Sojka, M. and Hanzalek, Z., 2006, "TORSCHE Scheduling Toolbox for Matlab", Proceedings of the IEEE International Symposium on Computer-Aided Control Systems Design, pp. 1181-1186.

Tipsuwan, Y. and Chow, M.Y., 2003, "Control Methodologies in Networked Control Systems", Control Engineering Practice, Vol. 11, No. 3, pp. 1099-1111.

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.

Zhang, W., Branicky, M.S. and Phillips, S.M., 2001, "Stability of Networked Control Systems", IEEE Control System Magazine, Vol. 21(1), pp. 84–99.

8. RESPONSIBILITY NOTICE

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