An Integrated Approach to Real-time Distributed Control Systems Over Fieldbuses

Pau Martí and Josep M. Fuertes
Dept. of Automatic Control
Universitat Politècnica de Catalunya
C/Pau Gargallo 5, 08028 Barcelona
Spain

Gerhard Fohler
Dept. of Computer Engineering
Mälardalen University
PO Box 883, SE-42123 Västerås
Sweden

Abstract – **The increasing application of flexible and powerful real-time distributed control systems is presently characterizing the industrial automation field. Such systems involve three main disciplines: control systems, real-time systems, and communication systems. Control systems, due their stringent timing constraints, demand real-time computing technology. In addition, communication systems are needed for the data messaging between field devices. In this paper, we propose an integrated approach to the design and implementation of such systems. We will show that by a separate control design and its posterior distributed implementation, the system performance may suffer degradation. That is, when control loops are closed over communication networks, timing problems, as communication induced varying delays, can appear, decreasing the control system performance, and even leading the system to instability. However, we will show that by an adequate integrated approach, that takes advantage of control theory, real-time communication properties, an adequate timing analysis, and an appropriate distribution of the control functions (architectural aspects), the system performance increases dramatically.**

I. INTRODUCTION

In the automation context, a real-time distributed control system is typically implemented by a set of computational devices (sensors, actuators, controllers, etc) that run one or several tasks, which communicate data across a field level communication network (fieldbus). The successful design and implementation of real-time distributed control application requires an appropriate integration of several disciplines including control systems, real-time systems and communication systems.

In this scenario, the key for distributed control systems is that almost no local control action can be taken in isolation from the rest of the system. As a consequence, control loops over communication networks are of main concern. The main parts of a control loop are sampling, control computation, and actuation. Control theory assumes that sampling should be performed at the same instant every period, control computation should start and finish quickly after the sample is available, and actuation should occur immediately after the control computation, or at a fixed instant after the sampling (depending on how the controller was designed).

That is, in control theory, the three main parts of a control loop are assumed to be instantaneous [2]. However, when a control loop is distributed implemented with several field devices that exchange data over fieldbus communication networks, at run time, such control loop timing assumptions are not met due to timing problems, leading to violations that can cause degradation in control performance and even instability.

Therefore, in order to achieve a successful implementation of real-time distributed control systems, it is imperatively important to derive and model all the timing constraints that the application must meet [13]. For instance, the network can cause time-varying delays in the communication [15] between field devices that can degrade the system performance, or the control computations competing for shared resources (scheduling problem) can cause unacceptable jitters for the control purposes, as it can be seen in [7].

However, applying real-time systems methodologies, those time-variations can be assessed (determined or, at least, bounded). For example, with respect to fieldbus communication, a formal analysis and suitable methodologies have been presented in [14] with the aim of guaranteeing before run-time that real-time distributed control systems can be successfully implemented with standard fieldbus communication networks.

We have to point out that in this paper, rather than analyzing and evaluating existing fieldbus communication systems for distributed control applications, we use general real-time communication properties in order to discuss the integrated approach we present.

Similarly to [14], since the early work in real-time scheduling [6], many results have been presented where timing analysis is the main concern. A common feature is that, according to worst-case assumptions, bounded time responses of tasks competing for shared resources are determined. For extensive revisions on real-time scheduling, see for example [3] and [4].

In this paper, we will show that by a separate control design and its posterior distributed implementation, the control performance of the distributed system may suffer significant degradation. However, with an adequate integrated approach to design and implementation of real-time distributed control systems, that takes advantage of modern control theory, real-time communication properties and analysis, and an appropriate distribution of the control functions (architectural aspects), the system performance increases dramatically.

Lately, several works have appeared presenting different approaches to integrated designs of real-time control systems, as in [1], [10] and [12]. However, no one of them takes into account architectural aspects and distribution of control procedures, which we'll show to be determinant for a successful implementation of real-time distributed control applications.

The rest of the paper is structured as follows: in section II, the system model and the problem statement are described. In section III, the implementation of current control models for control loops closed over communication networks is discussed and some implementation problems are outlined. Section IV presents the integrated approach to the design an implementation of real-time distributed control systems. Finally, in section V, conclusions and future work are discussed.

II. SYSTEM MODEL AND PROBLEM DESCRIPTION

We investigate an integrated approach to the design and implementation of control loops closed over fieldbus communication. Henceforth we will use an inverted pendulum mounted on a motor driven cart as a controlled plant example. A sketch of this system is shown in Fig. 1. However, the result we present is a general approach that, with the appropriate analysis, can be applied to other control problems.

The inverted pendulum control problem can be stated as follows: the inverted pendulum (of length *l* and mass *m*) can only swing in a vertical plane parallel to the direction of the cart (of mass *M*). To balance the pendulum, the cart is pushed back and forth on a track of limited length. Balancing fails when the inclination of the pendulum exceeds pre-set limits, or when the cart hits the stops at the end of the track. The aim is to find a controller to balance the inverted pendulum, preventing it from failing, and to bring the car to the center of the track.

The state of the inverted pendulum on a cart is described by the cart position (x), its velocity (v), the pendulum angle (θ) and the angular velocity (ω). The force provided to the cart (*u(t)*) is the controlling action (output), calculated according to the actual angle and position (*y(t)*, inputs). The approximate linearized time invariant state space form of the inverted pendulum that we will use in the control design is described in (1) and (2), where g is the gravity.

$$\begin{bmatrix} \dot{\theta} \\ \dot{\omega} \\ \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{(M+m)\cdot g}{M\cdot l} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{m\cdot g}{M} & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \theta \\ \omega \\ x \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{1}{M\cdot l} \\ 0 \\ \frac{1}{M} \end{bmatrix} u(t) \quad (1)$$

$$y(t) = \begin{bmatrix} \theta \\ x \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \theta \\ \omega \\ x \\ v \end{bmatrix} \quad (2)$$

For the sake of simplicity, we will focus only on the angle (θ). Therefore, the goal of our controller is to maintain the desired vertically oriented position of the inverted pendulum at all the times. It can be easily studied that the open loop system is highly unstable.

To start with the analysis, the control loop is implemented in a distributed architecture, with three nodes communicating across a fieldbus communication network, as in Fig. 2.

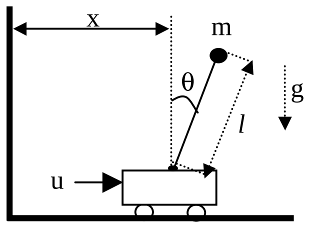

Fig.1 Inverted pendulum

A sensor node, strictly periodically (h, sampling period), samples the system (*y(t)*) and sends the data to the controller node, introducing a communication delay (τ_{sc}, sensor to controller delay). A controller node, that executes a single control computation that implements a suitable control design, introduces a computation delay (τ_c), that we assume constant for each controller execution. When the output is produced (*u(t)*), it is sent to the actuator node, introducing again another communication delay (τ_{ca}, controller to actuator delay). For simplicity purposes, we assume all necessary AD and DA conversions to be instantaneous. However, the timing analysis we are going to present can be easily extended taking into account these conversions as extra delays.

We have to point out that if the constant computation delay assumption is relaxed, the results we present still hold. In the last section will come back to this issue. However, from now, to simplify the reasoning, we assume a constant computation delay.

Given the architecture of our fieldbus-based real-time distributed control system, from a control point of view, what is important to assure is that samples are periodically taken and actuation is performed at equidistant time instants. In the architecture portrayed in Fig. 2, it is clear that, even the sampling is periodically done, the actuation won't be periodic due to the communication induced varying delays. That is, from sampling to actuation, varying delays will apply, depending on the fieldbus characteristics. Therefore, if the controller design is performed in the design stage without taking into account the posterior distributed implementation, the control system performance will suffer significant degradation.

For illustrative purposes, using the inverted pendulum and the linear time invariant state space form described by (1) and (2), we have obtained an appropriate state feedback control law (with the adequate sampling period h) through pole placement observer design approach that meets the specified closed loop requirements. However, in the state feedback control law, we haven't taken into account any possible implementation induced delays.

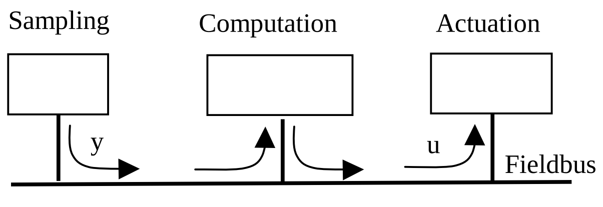

Fig.2 Fieldbus-based distributed architecture

Figure 3 shows a simulation of the ideal system response (angle) when the implementation of the system is free of delays. Henceforth, all the simulations have been done using the simulator presented in [5].

However, if the same control design is executed on the controller node of our distributed architecture, due to varying communication delays, the system performance decreases. Fig. 4 shows the effects of those delays in the controlled system response, for randomly induced communication delays of different magnitudes (approximately, 10%, 15%, 20% and 25% of the nominal sampling period h). With the aim of avoiding this degradation, what we firstly propose is to integrate the controller design with the distributed architecture, that is, to include all varying communication delays, which our distributed architecture introduces, in the design stage of the controller.

III. IMPLEMENTATION OF CONTROL MODELS FOR NETWORKED INDUCED DELAYS

In this section, we will discuss the feasibility of implementing current control models for networked induced delays [15], which are models that apply in our architecture. In this architecture, the samples are periodically taken, and after several induced delays, the actuation is performed.

Therefore, the controller has to be designed taking into account those delays that appears from the data acquisition (sampling) up to the actuation. It has to be pointed out that, from a control viewpoint, the computational delay of the control algorithm that is executing the controller node has not to be specifically treated, but taken into account in the controller design as another delay. The accumulation of all distributed architecture induced delays, called sampling-actuation delay, is described in (3).

$$\tau = \tau_{sc} + \tau_c + \tau_{ca} \tag{3}$$

For control purposes, a state-space model for the continuous time invariant system can be described by (4) and (5), including a constant delay τ, as in [2].

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t-\tau) \tag{4}$$

$$y(t) = Cx(t) + Du(t) \tag{5}$$

where we assume that τ is less than the sampling period. The sampled data system in a state-space from can be described then by (6) and (7).

$$x(kh+h) = \Phi(h)x(kh) + \Gamma_0(h,\tau)u(kh) + \Gamma_1(h,\tau)u(kh-h) \tag{6}$$

$$y(kh) = Cx(kh) + Du(kh) \tag{7}$$

where
$$\Phi(h) = e^{Ah}$$
$$\Gamma_0(h,\tau) = \int_0^{h-\tau} e^{As}\,ds B$$
$$\Gamma_1(h,\tau) = e^{A(h-\tau)} \int_0^{\tau} e^{As}\,ds B$$

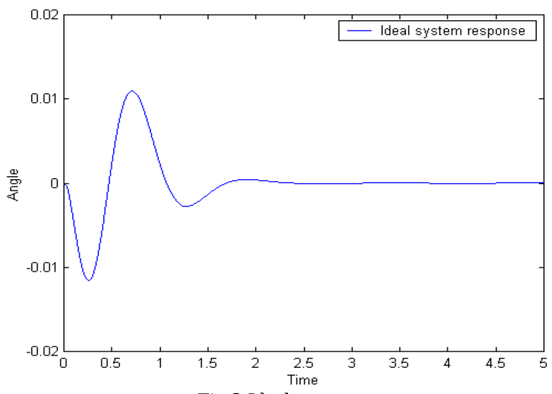

Fig.3 Ideal system response

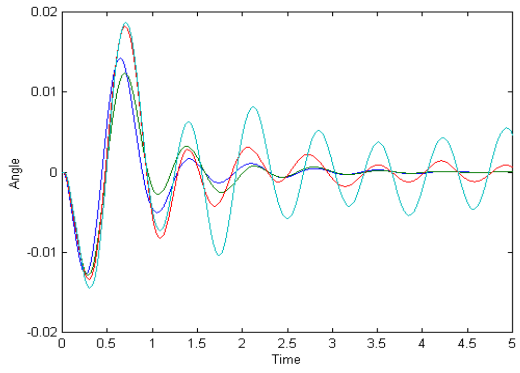

Fig.4 Degraded system response due to varying time-delays

To meet the closed loop system performance requirements according to the process dynamics, the system specified by (6) and (7) can be controlled using state feedback, expressed in (8), where L, the state feedback control law, can be obtained by an appropriate control design strategy such as pole placement or optimization approach.

$$u(kh) = -L(h,\tau)x(kh) \tag{8}$$

At the end, after all the design process, the controller node will be executing a control algorithm that will depend on the sampling period (h) and on constant sampling-actuation delay (τ).

Two implementation problems arise. Firstly, notice that while we assumed τ_c to be constant (one task executing in isolation on a single CPU), τ_{sc} and τ_{ca} will vary depending on the characteristics of the fieldbus communication, the medium access control policy and the data traffic. Therefore, the sampling-actuation delays will vary at each control loop execution. However, the state feedback control law in (8) obtained from (6) and (7) has been designed assuming a constant delay τ.

Secondly, even if the system in (6) and (7) still holds for time-varying delays, as it is suggested in [15], another problem arises: information availability. The control algorithm mandates that the sampling period and the sampling-actuation delays must be known at the beginning of each controller execution.

The sampling period is chosen at the design stage, and, since in our architecture we assumed that the sampler node is performing strictly periodic sampling, h is known and constant.

Moreover, since computation delay τ_c was assumed to be constant, it is also known at the beginning of each controller execution.

However, the varying sampling-actuation delays introduced by the communication network cannot be completely known at the beginning of each controller execution. At the beginning of each controller execution, sensor to controller delay, τ_{sc} can be known for example using time-stamping techniques if we assume a global time for clock synchronization in our architecture. We have to point out that assuming that clock synchronization will add overhead on the fieldbus traffic (and possibly additional hardware will be required), the extra load will only imply to have longer communication delays. Therefore, the approach we present will still hold because it does not depend on the delay duration.

Unfortunately, controller to actuator delay, τ_{sc}, will not be known at the beginning of each controller execution. The only thing we can guarantee is that this later delay will be bounded, that is, it will take values within a known rank, because real-time fieldbus communication assures bounded message latencies [14].

Therefore, at the design stage, what we can assure is whether the implementation can meet the timing constraints that the distributed application mandates. However, since one of the varying architectural introduced delays can not be known at the beginning of each controller execution, the system will still suffer significant degradation. Therefore, timing implementation details will preclude a successful execution of the controller even designed taking into account time delays.

IV. AN INTEGRATED APPROACH

In the previous section, we outlined the need of integrating the controller design with architectural characteristics of the physical distribution, in order to take into account, at the design stage, all fieldbus communication induced delays, that were shown to have a decisive influence on the control system performance (Fig. 4). However, we discussed that the fully applicability of the proposed control design, described by (6), (7) and (8), due to implementation details, was not possible. Therefore, we need still, a better approach.

We propose to integrate at the design stage, the controller design with an adequate architectural timing analysis of the distributed design, analyzing then wheter the current available control methods can be fully applied. If not (which is the example we were discussing in the last section), we propose to redesign the distributed architecture, taking into account what are the timing constraints that the control method mandates in its implementation.

In our case, the implementation of the control method we are dealing with (sampled-data system with time-delays) mandates that the sampling period (h) and the sampling-actuation delays (τ) must be known at the beginning of each controller execution.

Therefore, in order to meet this implementation timing constraint, what we need to know at the beginning of each controller execution is the exact sampling-actuation delay. This means for example, that if the controller is executed immediately before the output is sent to the plant, we will have the information we need. To achieve this requirement, what we can do is to change the architecture of our distributed control loop. If we move the control computation from the controller node to the actuator node and remove the controller node, as in Fig. 5, the problem is solved.

In this new architecture, the sensor node, that strictly periodically (h, sampling period) samples the system (*y(t)*), sends the data to the controller node, introducing a communication delay (τ_{sa}, sensor to actuator delay). The actuator node then, executes the single control computation, introducing a computation delay (τ_c), that we still assume constant for each controller execution. When the output is produced (u), it is immediately sent to the plant.

In this new architecture, each sampling-actuation delay can be known at the beginning of each control computation (using for example time-stamping techniques), fulfilling the timing constraint that the implementation of the controller design required. However, further analysis is needed in the controller design, because it was designed for constant time-delays and in our new architecture we are suffering varying but known time-delays. The new sampling-actuation delays will vary at each control loop execution because, even knowing that τ_c is fixed, τ_{sa} will vary due to fieldbus communication characteristics and dynamics. Notice that the sampling-actuation delay of the new architecture is represented by (9).

$$\tau = \tau_{sa} + \tau_c \tag{9}$$

Thus, the control design strategy must take into account this variation. What we know is that this variation will be limited, that is, τ_{sa} will take values within a known discrete rank, that can be determined at the design stage (offline) according to the fieldbus characteristics that is being used, in other words, according to the worst-case message latency. Therefore, at each k-execution of the control computation, the sampling-actuation delay (τ_k) will take values within the generalized rank described by (10).

$$0 \le \tau_k \le \text{Fieldbus-worst-case-message-latency} \tag{10}$$

Knowing this, the control computation can compensate at run time for each delay at each control loop execution if it implements the sampled-data system with time-delays described in (6) and (7) formulated as (11) and (12), where τ_k varies from execution to execution.

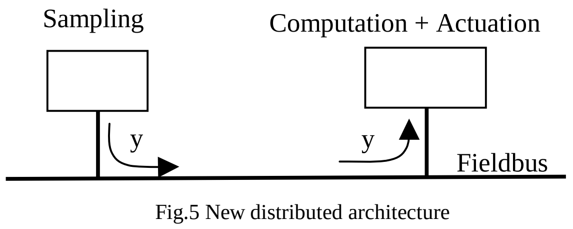

Fig.5 New distributed architecture

$$x(kh+h) = \Phi(h)x(kh) + \Gamma_0(h,\tau_k)u(kh) + \Gamma_1(h,\tau_k)u(kh-h) \quad (11)$$

$$y(kh) = Cx(kh) + Du(kh) \quad (12)$$

where
$$\Phi(h) = e^{Ah}$$
$$\Gamma_0(h,\tau_k) = \int_0^{h-\tau_k} e^{As} ds B$$
$$\Gamma_1(h,\tau_k) = e^{A(h-\tau_k)} \int_0^{\tau_k} e^{As} ds B$$

Since τ_k has a bounded variability due to real-time communication properties, the stability of the system can be determined as in [8]. In Fig. 6 we show a comparison of a simulation of the ideal inverted pendulum angle response (dashed line, same as in Fig. 3) and a simulation of the inverted pendulum angle response if the system is implemented according to the new architecture portrayed in Fig. 5 and according to the proposed sampled-data system with varying time-delays model described by (11) and (12) with the appropriate controller design that compensates for each sampling-actuation delay at each control computation (full line).

In Fig. 6 it can be seen that even the system response with delay compensation is not exactly the same as the ideal system response (due to the communication induced varying-delays), the improvement respect to the degraded system response (Fig. 4) is fairly clear.

Although we considered the new computation delay τ_c of the actuator node in the new architecture to be constant, we have to point out that the computational cost of the new control algorithm implementing the control law for the sampled-data with varying time-delays described by (11) and (12) will add a computational overhead to the initial control algorithm implementing the control law for the model with constant time-delays described by (6) and (7). However, even the new delay will be longer than the previous one, its degrading effects on the performance will be also compensated using the presented approach because its successful applicability does not depend on the delay duration.

This additional computation overhead will depend on the design method and on the controller strategy used. However, since in the actuator node, the control computation is the only task to be executed, its computation time can be exactly determined.

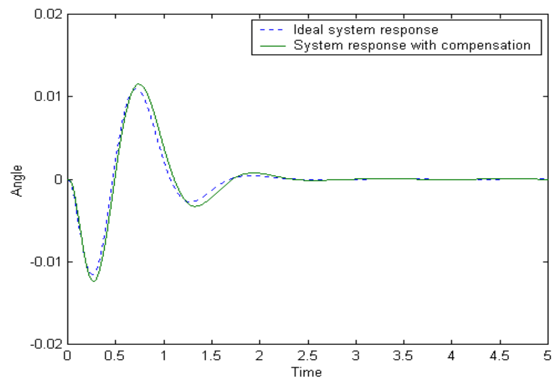

Fig.6 Ideal vs. compensated system response

Even more, if in the actuator node several tasks (control and non control tasks) are competing for the CPU as in a usual real-time implementation, the control algorithm computation delay may not been known at the beginning of each control computation due to jitters that current real-time scheduling algorithms introduce in task instance executions. Therefore, the applicability of the presented approach wouldn't be, a priori, possible.

However, applying real-time novel techniques [9], even in this case, the applicability of the integrated approach we have presented in this paper is feasible. That is, these jitters introduced by the scheduling algorithm can be, firstly, offline determined, and secondly, online compensated with the appropriate controller parameters adjustment.

V. CONCLUSIONS

In this paper we have presented an integrated approach to the design and implementation of real-time fieldbus-based distributed control systems. Firstly, we have shown that by a separate controller design and its posterior implementation in a fieldbus-based distributed architecture, the performance of the controlled systems suffers significant degradation.

Afterwards, we have shown that even trying to take into account architectural characteristics in the control design stage (that is, fieldbus communication induced delays) specific timing requirements that the implementation of the control design mandates, precludes a successful implementation of the distributed application.

We have presented an integrated approach to overcome this problem such that, apart from taking into account architectural characteristics in the control design stage, implies architecture and control co-design, so as to achieve a successful implementation of the whole real-time fieldbus-based distributed control system.

Further research will focus on how to characterize the quality of service, in terms of control performance and delays, of the control computation implementing the new control strategy in the new architecture. Using then this quality of service characterization, we will be able to assess other distributed architectures along with other control strategies.

VI. ACKNOWLEDGEMENTS

The presented research has received support from Spanish CICYT project ref. DPI2000-1760-C03-01.

VII. REFERENCES

[1] K.-E. Årzen, B. Bernhardsson, J. Eker, A. Cervin, K. Nilsson, P. Persson and L. Sha, "Integrated Control and Scheduling" Research report ISSN 0280-5316, ISRN LUTFD2/TFRT7586-SE August, 1999

[2] K.J. Åström and B. Wittenmark. *Computer-Controlled Systems. Third edition*. Prentice Hall. 1997

[3] N.C. Audsley, A. Burns, R.I. Davis, K.W. Tindell and A.J. Wellings, "Fixed Priority Pre-emptive Scheduling:

An Historical Perspective". *Journal of Real-Time Systems*, 8, 173-198. 1995

[4] G. Buttazzo. *Hard Real-Time Computing Systems. Predictable Scheduling Algorithms and Application*. Kluiwer Academic Publishers, 1997

[5] J.Eker and A. Cervin, "A Matlab Toolbox for Real-Time and Control Systems Co-Design", *in Proc. of the 6th Int. Conference on Real-Time Computing Systems and Applications,* Hong Kong, China, December 1999

[6] C. Liu and J. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment", *J.ACM,* 20, 46-61. 1973

[7] P. Marti, R. Villa, J.M. Fuertes and G. Fohler, "On Real-Time Control Tasks Schedulability", *IFAC European Control Conference,* September, 2001

[8] P. Marti, R. Villa, J.M. Fuertes and G. Fohler, "Stability of On-line Compensated Real-Time Scheduled Control Tasks", *Accepted for publication to IFAC Conference on New Technologies for Computer Control,* November, 2001

[9] P. Marti, G. Fohler, K. Ramamritham and J.M. Fuertes, "Jitter Compensation in Real-Time Control Systems" *Accepted for publication to 22th IEEE Real-Time Systems Symposium*, December, 2001

[10] H.Rehbinder, M. Sanfridson, "Integration of Off-Line Scheduling and Optimal Control", *12th Euromicro Conference on Real-Time Systems*, Sweden, June 2000

[11] D. Seto, J.P. Lehoczky, L. Sha and D.G. Shin, "On task Schedulability in Real-Time Control Systems", *RT Systems Symposium, 17th IEEE.* p 13-21, 1996

[12] L. Sha, X. Liu, M. Caccamo and G. Buttazzo, "Online Control Optimization Using Load Driven Scheduling", *39th IEEE Conference on Decision and Control, Sydney*, Australia, December 12-15, 2000

[13] M. Törngren, "Fundamentals of Implementing Real-time Control Applications in Distributed Computer Systems". *J. of Real-Time Systems*, 14, 219-260, Kluwer Academic Publishers. 1998

[14] E. Tovar. "Supporting Real-Time Communications with Standard Factory-Floor Networks". Phd Dissertation, Porto University, 1999.

[15] B. Wittenmark., J. Nilsson and M. Törngren, "Timing Problems in Real-Time Control Systems". *Proc. Of the American Control Conference*. US. 1995