Centro de Sistemas Telemáticos e Computacionais

Instituto Superior Técnico

NavIST Group
Fault-Tolerant Real-Time Distributed
Systems and Industrial Automation

Fundamental Issues in the Design of a CAN-based Fault-Tolerant Real-Time Communication Infrastructure for DEAR-COTS

CSTC Technical Report RT-99-02

J. Rufino, P. Veríssimo, G. Arroz

October 1999

Centro de Sistemas Telemáticos e Computacionais - NavIST Group
Fault-Tolerant Real-Time Distributed Systems and Industrial Automation

Fundamental Issues in the Design of a CAN-based Fault-Tolerant Real-Time Communication Infrastructure for DEAR-COTS

Project PRAXIS/P/EEI/14187/1998 - DEAR-COTS
Distributed Embedded ARchitecture using Commercial Off-The-Shelf components

Technical Report: CSTC RT-99-02
Authors: J. Rufino, P. Veríssimo, G. Arroz
Date: October 1999

Avenida Rovisco Pais, 1049-001 Lisboa - PORTUGAL, Tel: +351-218418397/99, Fax: +351-218417499.
NavIST WWW Page URL - http://pandora.ist.utl.pt.

Fundamental Issues in the Design of a CAN-based Fault-Tolerant Real-Time Communication Infrastructure for DEAR-COTS*

José Rufino
ruf@digitais.ist.utl.pt
IST-UTL†

Paulo Veríssimo
pjv@di.fc.ul.pt
FC/UL‡

Guilherme Arroz
pcegsa@alfa.ist.utl.pt
IST-UTL

Abstract

Standard fieldbuses are today a cost-effective solution for distributed computer control systems. However, the efficient implementation of fault-tolerance and real-time mechanisms on the simple fieldbus environment presents non-negligible problems. This paper outlines the approach to be taken in the DEAR-COTS Project with regard the use of the Controller Area Network (CAN) as an off-the-shelf component in the design of fault-tolerant real-time distributed systems.

1 Introduction

The design and implementation of distributed computer control systems intended for real-world interfacing, i.e. integrating sensors and/or actuators, have increasingly been based on standard fieldbuses as an alternative to specialized and thus costly architectures [7]. The development of applications for such environments may greatly benefit from the availability of services such as clock synchronization, reliable group communication, membership and failure detection.

However, the migration of fault-tolerant communication systems to the realm of fieldbuses presents non-negligible problems, some of them addressed by our previous works in the context of CAN, the Controller Area Network [21, 16, 10, 13]. CAN is a fieldbus that has assumed increasing importance and widespread acceptance in application areas as diverse as shop-floor control, robotics or automotive [6]. The inherent reliability and real-time attributes of CAN, its low-cost and wide commercial availability, and a reasonable level of simplicity and flexibility in system design, are reasons that justify the interest of having the DEAR-COTS communication infrastructure build around the CAN fieldbus [3].

This paper outlines our approach on how to use CAN as an off-the-shelf component in the design of fault-tolerant real-time distributed systems: Section 2 analyzes CAN dependability; Section 3 defines the system model; CAN non-stop operation in the presence of medium failures is discussed in Section 4; Section 5 addresses CAN inaccessibility and its impact on hard real-time operation. The reliable group communication, failure detection and membership, and clock synchronization services are discussed in Sections 6 to 8. Finally, Section 9 concludes the paper.

2 Dependability of CAN

CAN uses a twisted pair cable as transmission medium. The CAN physical layer specified in [6] allows tolerance of some cabling faults (one wire open/short failures), by switching from differential

*This work was partially supported by FCT, through Project PRAXIS/P/EEI/14187/1998 (DEAR-COTS).

†Instituto Superior Técnico - Universidade Técnica de Lisboa, Avenida Rovisco Pais, 1049-001 Lisboa, Portugal. Tel: +351-218418397 - Fax: +351-218417499. NavIST Group CAN WWW Page - http://pandora.ist.utl.pt/CAN.

‡Faculdade de Ciências da Universidade de Lisboa, Campo Grande - Bloco C5, 1700 Lisboa, Portugal. Tel: +351-217500087 - Fax: +351-217500084. Navigators Home Page: http://www.navigators.di.fc.ul.pt.

to single-wire operation. However, this mechanism cannot provide resilience to the simultaneous interruption of both wires.

CAN is a multi-master fieldbus. Bus signaling takes one out of two values: *recessive*, otherwise the state of an idle bus; *dominant*, which always overwrites a recessive value. This behavior, together with the uniqueness of frame identifiers, is exploited for bus arbitration. A *carrier sense multi-access with deterministic collision resolution* policy is used. The node transmitting the frame with the lowest identifier always goes through and gets the bus. Frames that have lost arbitration or have been destroyed by errors are automatically retransmitted. A *frame* is a piece of encapsulated information traveling on the network. It may contain a *message*, a user-level piece of information.

Though CAN fault-confinement and error detection mechanisms ensure that most failures are perceived consistently by all nodes [12], some subtle errors can lead to inconsistency. Inconsistent frame omissions occur when faults hit the last but one bit of a frame at some recipients[1], tagged $\times$ set in Figure 1-B. This may lead to: the message to be accepted in duplicate by the recipients in the $\bullet$ set of Figure 1-B, upon retransmission; inconsistent message omission, if the sender fails before retransmission. A thorough discussion of these failure scenarios can be found in [16]. However infrequent they may be, the probability of its occurrence is high enough to be taken into account, at least for highly fault-tolerant applications of CAN.

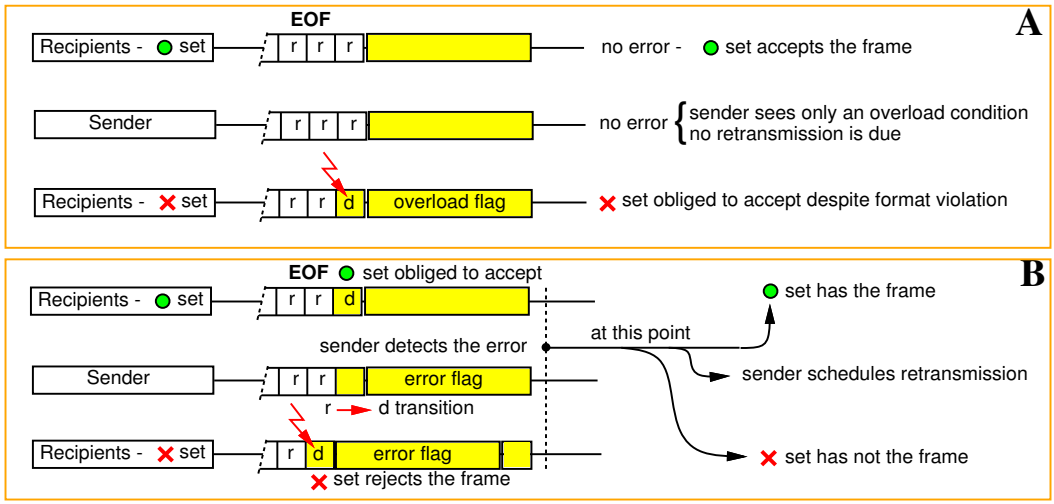

Figure 1: Inconsistency in CAN error handling

3 System Model

We enumerate our fault assumptions, formalizing the discussion of Section 2, and present the properties of our system model. The model addresses a set of processes communicating through CAN. Each process is attached to the network through a CAN interface. Together, they form a node. We assume that the processes are fail-silent and blame all temporary failures on the CAN network components. We say a component is **weak-fail-silent** if it behaves correctly or crashes if it does more than a given number of omission failures in a time interval of reference, called the component's *omission degree*. The **CAN network components** are modeled by the failure semantics used in [16]:

[1] The set may have only one element. Examples of causes for inconsistent detection are: electromagnetic interference or deficient receiver circuitry.

- individual components are **weak-fail-silent** with *omission degree* f_o;
- failure bursts never affect more than f_o transmissions in an interval of reference [2];
- omission failures may be inconsistent (i.e., not observed by all recipients);
- there is no permanent failure of shared network components, as justified in Section 4.

CAN MAC-level properties

CAN has a medium access control (MAC) sub-layer that basically exhibits the same kind of properties identified in previous works on LANs [19]. A first formalization of CAN MAC-level properties in [16] proved quite effective. Figure 2 complements that definition with the time-related properties MCAN5-MCAN7.

MCAN1 - Broadcast: correct nodes receiving an uncorrupted frame transmission, receive the same frame.

MCAN2 - Error Detection: correct nodes detect any corruption done by the network in a locally received frame.

MCAN3 - Network Order: any two frames received at any two correct nodes, are received in the same order at both nodes.

MCAN4 - Bounded Omission Degree: in a known time interval T_{rd}, omission failures may occur in at most k transmissions.

MCAN5 - Bounded Inaccessibility: in a known time interval T_{rd}, the network may be inaccessible at most i times, with a total duration of at most T_{ina}.

MCAN6 - Bounded Transmission Delay: any frame queued for transmission is transmitted on the network within a bounded delay of $T_{td} + T_{ina}$.

MCAN7 - Tightness: correct nodes receiving an uncorrupted frame transmission, receive it at real time values that differ, at most, by a known small constant $\Delta\Gamma_{tight}$.

Figure 2: CAN MAC-level properties

MCAN4 maps the failure semantics introduced earlier onto the operational assumptions of CAN, being $k \geq f_o$. MCAN6 specifies a maximum frame transmission delay, which is T_{td} in the absence of faults. It depends on message latency classes and offered load bounds [17, 22, 8, 1]. The *bounded transmission delay* includes T_{ina} (MCAN5), the maximum duration of an inaccessibility fault [21]. MCAN7 is crucial for achieving high precision on synchronized clocks [10].

CAN LLC-level properties

CAN has error-recovery mechanisms on top of the basic MAC sub-layer functionality, that yield interesting message properties. These mechanisms provide additional dependability guarantees, in some way with the flavor of the logical link control (LLC) sub-layer in LANs: the omission failures specified by MCAN4 are masked in general at the LLC level by the retry mechanism of CAN. However, the existence of inconsistent omissions, as discussed in Section 2, postulates:

- that there may be message duplicates when they are recovered;

[2] For instance the duration of a message transaction round. Note that this assumption is concerned with the total number of failures of possibly different components.

LCAN1 - Validity: if a correct node broadcasts a message, then the message is eventually delivered to a correct node.

LCAN2 - Best-effort Agreement: if a message is delivered to a correct node, then the message is eventually delivered to all correct nodes, if the sender remains correct.

LCAN3 - At-least-once Delivery: any message delivered to a correct node is delivered at least once.

LCAN4 - Non-triviality: any message delivered to a correct node was broadcast by a node.

LCAN5 - Total Order: *not ensured.*

LCAN6 - Bounded Inconsistent Omission Degree: in a known time interval T_{rd}, inconsistent omission failures may occur in at most j transmissions.

Figure 3: Native CAN LLC-level properties

- that some j of the k omissions ($j \ll k$) will show at the LLC interface as inconsistent omissions.

Figure 3 recalls from [16] the LLC-level properties of CAN. The first five properties characterize the reliability of CAN communication and its shortcomings. LCAN6 provides the grounds for the design of efficient dependability enforcement mechanisms [16, 11].

4 Network Availability

Our ideas to enhance CAN network availability, thoroughly discussed in [13, 14], are sketched in Figure 4. They rely on the replication of the physical path – cable medium and transceivers – used by the MAC entities to communicate (channel).

The strategy for channel media replication assumes: each cable replica is routed differently, being reasonable to consider failures in different media as independent; any bit issued from a MAC sub-layer is simultaneously transmitted on all the redundant media interfaces.

The bare CAN wired-AND nature is exploited for handling channel replicated media: the signals from the different redundant media receivers are combined in an AND function, before interfacing the MAC sub-layer. This *Columbus' egg* idea [13], constitutes a simple method to secure resilience to CAN physical partitions:

- nodes at the *in-partition*[3] receive a correct signal on all redundant media interfaces;
- in the *out-partition*, the recessive signal from the (idle) failed media is combined with the redundant media signals to produce a correct channel output.

However, to be of practical use such a scheme should be enhanced to support a less restrictive and thus more realistic fault model [13]: stuck-at-dominant faults can be handled through a special-purpose *watchdog* timer; omission faults can be monitored and a medium exhibiting an excessive number of omission errors can be put in *quarantine* until (and if) it "behaves well" again.

The detection of abnormal bus idle periods (recessive state) is relevant for high-level diagnose applications aiming to: distinguish a stuck-at-recessive from a medium partition failure; pinpoint the location of the medium partition failure in the network cable.

[3] I.e., the partition that includes the transmitter.

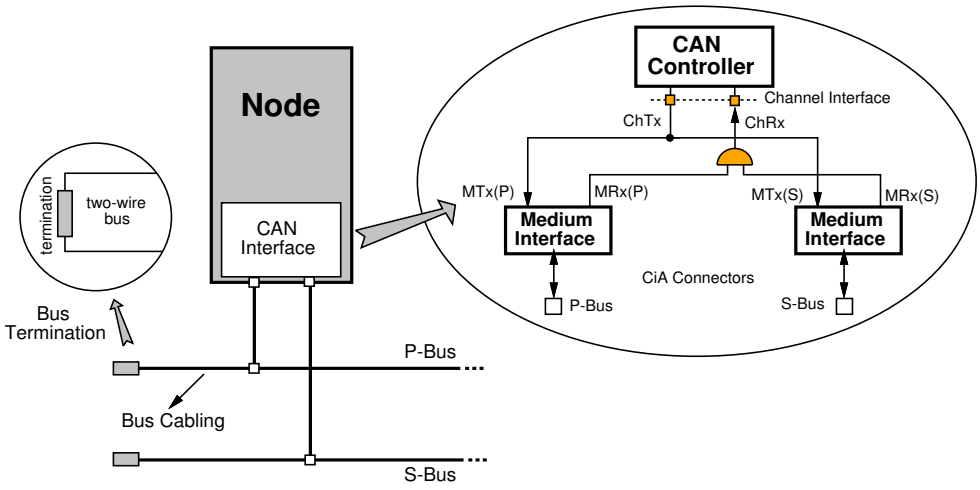

Figure 4: A *Columbus' egg* idea for bus media redundancy in CAN

5 CAN Inaccessibility

Even in a continuously connected network, the occurrence of certain events (e.g.: bit errors; receiver glitches) in its operation produces a subtle form of virtual partitioning. CAN has its own means of recovering from these situations, but this takes time [12, 21]. An **inaccessibility** fault occurs when a component *temporarily refrains* from providing service.

The effect of inaccessibility on real-time communication is the error it introduces in timing bounds, such as message latencies. Most message schedulability analyses consider the network as always functioning normally [17, 5, 18]. Bounds are established that may be violated upon the occurrence of inaccessibility events. In consequence, the system may exhibit an unpredictable behavior and ultimately fail. Examples of applications where the influence of the periods of inaccessibility has been completely disregarded can be found in [5, 18].

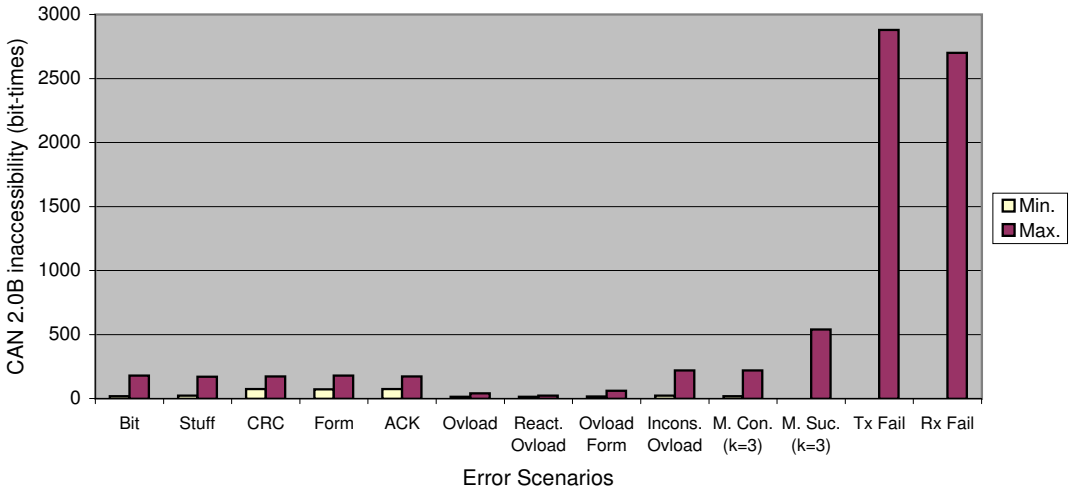

Figure 5: Normalized CAN inaccessibility

One solution to avoid failures due to inaccessibility events is: to compute the duration of all possible inaccessibility faults, as done in [12, 21] (Figure 5 summarizes the results); accommodate inaccessibility bounds in the timeliness model (e.g. MCAN6).

6 Reliable Group Communication

The problem of reliable CAN communication was originally addressed in [16], dismissing the current belief that CAN supports an atomic broadcast service and providing a protocol suite that handles the problem effectively. Next, we use and extend the results from [16] to outline the architecture specification of a CAN-based reliable group communication service.

Interfacing the standard CAN layer (Figure 6) we use the fundamental fault-tolerant broadcast protocols of [16, 11]: SDCAN enhances LCAN3, by ensuring that each message is delivered **at-most-once**, if no message ordering is required [11]; EDCAN enhances LCAN3 in the same way but also enhances LCAN2, removing the condition of the sender not failing (cf. Figure 3) and securing all the properties of a reliable broadcast service [16, 11].

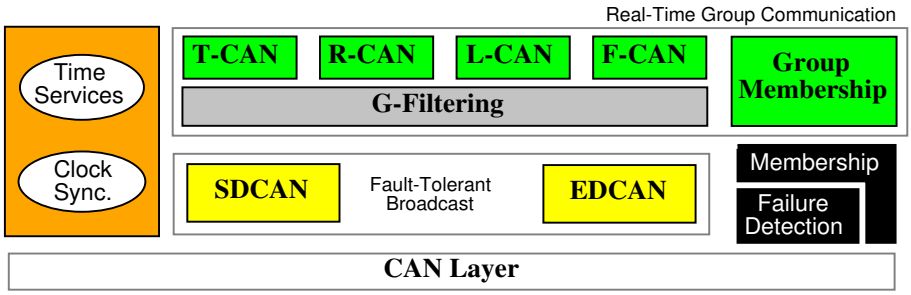

Figure 6: CAN real-time fault-tolerant protocol suite

A versatile real-time group communication service, offering different *qualities of service*, is defined above this layer. The *G-Filtering* sub-layer restricts processing of higher layers to the traffic addressed to the node. The top sub-layer include (totally ordered) atomic (T-CAN) and reliable (R-CAN) group communication protocols, which are variants of the protocols in [16], and two new protocols: L-CAN, a reliable group communication protocol that trades a high message delivery bound with a low utilization of network bandwidth; F-CAN, a companion protocol that exploits MCAN3 and LCAN2 to support an efficient message fragmentation scheme that does not need to use sequence numbers for fragment ordering.

The failure detection and membership protocols, discussed next, are also included in the architecture specified in Figure 6.

7 Failure Detection and Membership

A membership service is intended to provide, at any given time, consistent information about failed/correct nodes. Our approach to this problem is based on the observation that many CAN applications [17, 5] exhibit a periodic traffic pattern. A failure detection/membership service matching strict application-level latency requirements can be designed with minimal costs in network bandwidth.

The periodic high-level messages are implicitly used as heartbeats (Figure 7). Specific *life-sign* messages need only to be issued by nodes with message periods higher than the failure detection latency or transmitting only sporadic/aperiodic traffic. If a node remains silent during a period longer than the detection latency, that will be a failure. Consistency of membership information is ensured through: a *reception history agreement* (RHA) protocol, in the presence of node join/leave operations; an optimized *failure detection agreement* (FDA) protocol, upon node failure.

Optimization tradeoffs exist with a protocol variant where all nodes explicitly issue *life-sign* messages.

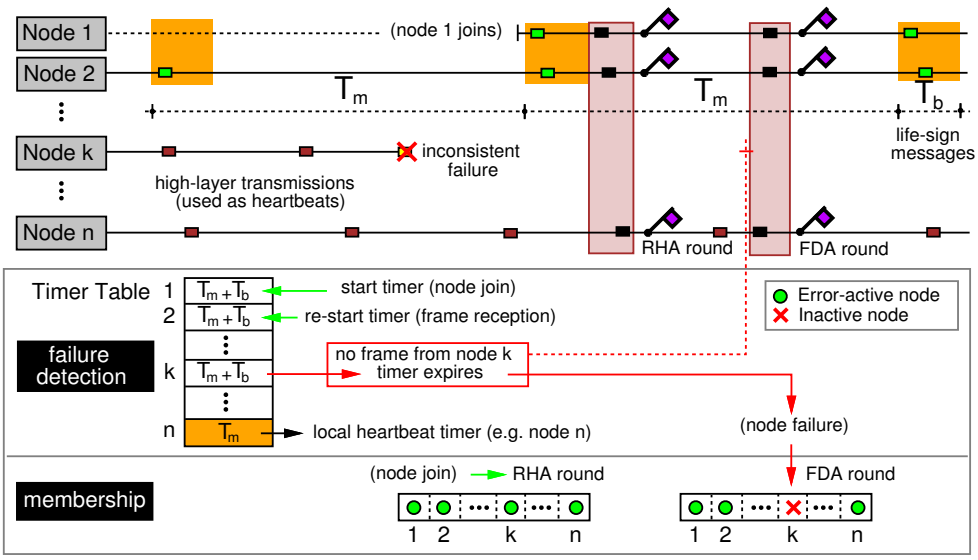

Figure 7: Failure detection and membership

8 Clock Synchronization

The aim of a clock synchronization service is to provide all correct processes of the system with a global timebase, despite the occurrence of faults in the network infrastructure or in a minority of processes. A common approach is to use the node hardware clock to create a virtual clock, which is locally read. All virtual clocks are internally synchronized by a *clock synchronization algorithm*.

In [10], it is described a clock synchronization algorithm inspired of the generic *a posteriori agreement* algorithm for broadcast networks [20] and of a non fault-tolerant CAN clock synchronization algorithm [4]. Significantly different from those algorithms, the new protocol was dubbed *phase-decoupled* and explicitly exploits the CAN properties to offer a clock synchronization service with a tight precision and a good accuracy, at reasonable bandwidth costs.

A *hierarchical* approach can be used to combine internal and external clock synchronization and to synchronize several CAN network segments, by making use of the techniques described in [20] to provide clock synchronization beyond the borders of a single broadcast segment.

9 Conclusions and Future Work

There is an increasing demand for fault-tolerant and real-time distributed systems based on fieldbuses. In this paper, after formalizing the properties actually secured by CAN, we have shown that with the appropriate techniques one can draw a modular solution able to add fault-tolerance and hard real-time attributes to the basic functionality offered by CAN off-the-shelf components.

This paper represents a first contribution to the definition and design of a CAN-based fault-tolerant real-time communication infrastructure for DEAR-COTS [3]. We have paid particular attention to dependability concerns, given our ongoing research work [21, 16, 10, 13].

Future project activities should address timeliness properties, in order to ensure the real-time requirements of the DEAR-COTS architecture are fulfilled [9, 3]. There have been published works addressing the real-time behavior of CAN [17, 22, 8, 1]. However, with a few exceptions [17, 8], those analysis are based on no-fault scenarios. To characterize the influence of the CAN dependability constraints in the timeliness properties of a CAN-based system, it is required to study

the system under a performability perspective. Furthermore, such an analysis should be extended to all relevant levels of the system [2, 19]. This is related to [15]:

- the calculation of the real worst-case message transmission delays (having in mind the analysis of message worst-case latencies and network schedulability);
- the calculation of the real worst-case protocol execution times (for real-time processing purposes);
- the dimensioning of timeouts used in the surveillance of remote interactions between peer entities.

References

[1] L. Almeida, J. Fonseca, and P. Fonseca. A flexible time-triggered communication system based on the Controller Area Network: Experimental results. In D. Dietrich, P. Neumann, and H. Schweinzer, editors, *Fieldbus Technology - System Integration, Networking and Engineering*, pages 342–350. Springer, September 1999. (Proceedings of the Fieldbus Conference FeT'99, Magdeburg, Germany).

[2] I. Blum and G. Juanole. Comparing the networks CAN and ARINC 629 CP with respect to the quality of service provided to an automatic control application. In D. Dietrich, P. Neumann, and H. Schweinzer, editors, *Fieldbus Technology - System Integration, Networking and Engineering*, pages 128–135. Springer, September 1999. (Proceedings of the Fieldbus Conference FeT'99, Magdeburg, Germany).

[3] DEAR-COTS Consortium. DEAR-COTS: a Distributed Embedded ARchitecture using Commercial Off-The-Shelf components - Technical Annex. PRAXIS XXI Program - Project Proposal, July 1998.

[4] M. Gergeleit and H. Streich. Implementing a distributed high-resolution real-time clock using the CAN-bus. In *Proceedings of the 1st International CAN Conference*, pages 9.02–9.07, Mainz, Germany, September 1994. CiA.

[5] J. Gil, A. Pont, G. Benet, J. Blanes, and M. Martinez. A CAN architecture for an intelligent mobile robot. In *Proceedings of the IFAC International Symposium on Intelligent in Components and Instrumentation for Control Applications*, Annecy, France, 1997. IFAC.

[6] ISO. *International Standard 11898 - Road vehicles - Interchange of digital information - Controller Area Network for high-speed communication*, November 1993.

[7] H. Kopetz and G. Grunsteidl. TTP - a protocol for fault-tolerant real-time systems. *IEEE Computer*, 27(1):14–23, January 1994.

[8] M.A. Livani, J. Kaiser, and W.J. Jia. Scheduling hard and soft real-time communication in the controller area network (CAN). In *Proceedings of the 23rd IFAC/IFIP Workshop on Real-Time Programming*, Shantou, China, June 1998. IFAC/IFIP.

[9] L. Pinho. The DEAR-COTS hard real-time subsystem. In *Proceedings of the 1st DEAR-COTS Workshop*, Porto, Portugal, October 1999. DEAR-COTS Consortium. (presentation slides).

[10] L. Rodrigues, M. Guimarães, and J. Rufino. Fault-tolerant clock syncronization in CAN. In *Proceedings of the 19th Real-Time Systems Symposium*, Madrid, Spain, December 1998. IEEE.

[11] J. Rufino, N. Pedrosa, J. Monteiro, P. Veríssimo, and G. Arroz. Hardware support for CAN fault-tolerant communication. In *Proceedings of the 5th International Conference on Electronics, Circuits and Systems*, pages 263–266, Lisbon, Portugal, September 1998. IEEE.

[12] J. Rufino and P. Veríssimo. A study on the inaccessibility characteristics of the Controller Area Network. In *Proceedings of the 2nd International CAN Conference*, pages 7.12–7.21, London, England, October 1995. CiA.

[13] J. Rufino, P. Veríssimo, and G. Arroz. A Columbus' egg idea for CAN media redundancy. In *Digest of Papers, The 29th International Symposium on Fault-Tolerant Computing Systems*, pages 286–293, Madison, Wisconsin - USA, June 1999. IEEE.

[14] J. Rufino, P. Veríssimo, and G. Arroz. Design of bus media redundancy in CAN. In D. Dietrich, P. Neumann, and H. Schweinzer, editors, *Fieldbus Technology - System Integration, Networking and Engineering*, pages 375–380. Springer, September 1999. (Proceedings of the Fieldbus Conference FeT'99, Magdeburg, Germany).

[15] J. Rufino, P. Veríssimo, and G. Arroz. Reliable real-time communication on CAN. In *Proceedings of the 1st DEAR-COTS Workshop*, Porto, Portugal, October 1999. DEAR-COTS Consortium. (presentation slides).

[16] J. Rufino, P. Veríssimo, G. Arroz, C. Almeida, and L. Rodrigues. Fault-tolerant broadcasts in CAN. In *Digest of Papers, The 28th International Symposium on Fault-Tolerant Computing Systems*, pages 150–159, Munich, Germany, June 1998. IEEE.

[17] K. Tindell and A. Burns. Guaranteeing message latencies on Controler Area Network. In *Proceedings of the 1st International CAN Conference*, pages 1.2–1.11, Mainz, Germany, September 1994. CiA.

[18] P. Tuominen and T. Virvalo. Synchronization of servosystem using CAN. In *Proceedings of the 2nd International CAN Conference*, pages 9.12–9.20, London, England, October 1995. CiA.

[19] P. Veríssimo. Real-time Communication. In S.J. Mullender, editor, *Distributed Systems*, ACM-Press, chapter 17, pages 447–490. Addison-Wesley, 2nd edition, 1993.

[20] P. Veríssimo, L. Rodrigues, and A. Casimiro. Cesiumspray: a precise and accurate global time service for large-scale systems. *Journal of Real-Time Systems*, 12(3):243–294, 1997.

[21] P. Veríssimo, J. Rufino, and L. Ming. How hard is hard real-time communication on field-buses? In *Digest of Papers, The 27th International Symposium on Fault-Tolerant Computing Systems*, Washington - USA, June 1997. IEEE.

[22] K. Zuberi and K. Shin. Scheduling messages on Controller Area Network for real-time CIM applications. *IEEE Transactions on Robotics and Automation*, 13(2):310–314, April 1997.