Embedded Platforms for Distributed Real-Time Computing: Challenges and Results

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

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

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

Abstract

Object-oriented techniques have been along the last decade one of the most useful programming paradigms. However, for distributed embedded systems the semantic gap between the object-orientation layer and the underlying infrastructure is extremely large. This gap can be narrowed, should the embedded system platform provide semantically rich communication and management services. This paper outlines our research effort in the provision of such services by CAN-based (Controller Area Network) systems.

1 Introduction

Object-oriented techniques have been along the last decade one of the most useful programming paradigms. Exhibiting a set of interesting characteristics, such as data abstraction and encapsulation, or inheritance and re-utilization, the object-oriented techniques allow a modular and flexible approach to software design.

However, the combination of object-orientation with distribution, fault-tolerance and real-time requirements involves the resolution of non-trivial problems, which constitute an open and challenging field of research, regarding both the communication and processing environments. A particular aspect of this problem appears when dealing with embedded systems. Whilst objects and dependability have been addressed many often in large-scale systems, for embedded systems the semantic gap between the object-orientation layer and the embedded infrastructure (fieldbuses, real-time micro-kernels) is extremely large.

This semantic gap can be narrowed, should an embedded system platform provide semantically rich services such as group communication, failure detection and membership, clock synchronization and group management services. In our recent work, we have been addressing the design and implementation of those services, in the context of the Controller Area Network (CAN) [24, 20, 17, 16]. CAN is a fieldbus that has assumed increasing acceptance in application areas as diverse as shop-floor control, robotics or automotive [9]. This paper provides an overview of such research effort.

The paper is organized as follows: Section 2 provides a short description of CAN and analyzes CAN dependability and availability; Section 3 discusses the system model; Sections 4 to 6 present the CAN group communication, failure detection and membership, and clock synchronization services; some fundamental issues concerning the interaction of these services with a higher-level object orientation layer are discussed in Section 7; Section 8 concludes the paper.

2 Controller Area Network

CAN is a multi-master fieldbus that uses a twisted pair cable as transmission medium [9, 3]. The network maximum length depends on the data rate. Typical values are: 40m @ 1 Mbps; 1000m @ 50 kbps. 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. When several nodes compete for bus access, 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 scheduled for retransmission. A *frame* is a piece of encapsulated information traveling on the network. It may contain a *message*, a user-level piece of information.

CAN dependability and availability

CAN fault-confinement and error detection mechanisms ensure that most failures are perceived consistently by al-

* Instituto Superior Técnico - Universidade Técnica de Lisboa, Avenida Rovisco Pais, 1049-001 Lisboa, Portugal. Tel: +351-1-8418397/99 - Fax: +351-1-8417499. 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: +351-1-7500087 - Fax: +351-1-7500084. Navigators Home Page: http://www.navigators.di.fc.ul.pt.

l nodes [18]. Unfortunately, some subtle errors can lead to inconsistency and induce the failure of dependable communication protocols based on CAN operation alone. 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 [20]. 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.

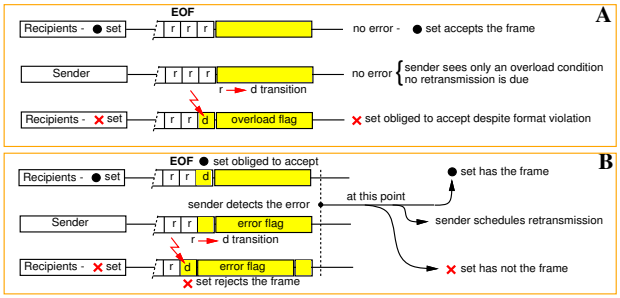

Figure 1. Inconsistency in CAN error handling

CAN is traditionally viewed as a robust fieldbus. The physical layer specified in [9] allows a few cabling faults (one wire open/short failures) to be tolerated, by switching from a two-wire differential operation to a single-wire mode [14]. However, no standardized mechanism exists to provide resilience against network partitioning if both wires of the network cable get simultaneously interrupted. Upon such a failure, there may be subsets of nodes which cannot communicate with each other. A solution to the problem has to be built as an extension to the standard specification.

3 System Model

The definition of a systemic model for our CAN-based system proved extremely useful: not only did it show the weaknesses of CAN with regard to communications reliability, but it provided the grounds to handle the problem effectively. Next, we enumerate our fault assumptions, formalizing the discussion of Section 2, and present the properties that underpin 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 introduce the following definition: a component is **weak-fail-silent** if it behaves correctly or crashes if it does more than a given number of omission failures in an interval of reference, called the component's *omission degree*. This assumption can be enforced by resorting to CAN own error confinement mechanisms, as explained in [20]. As a consequence, we have defined the following failure semantics for the **CAN network components** [20, 19]:

- 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 the communication **channel**.

We call **channel** to the physical path – cable medium and transceivers – used by the MAC[3] entities to communicate. Resilience against *all* the cabling failures discussed in Section 2 can be enforced through channel redundancy. An existing commercial solution implements a self-healing ring/bus architecture [13], but does not solve the problem of CAN continuity of service efficiently: ring reconfiguration takes time (it can last as long as 100 ms) and meanwhile the network is partitioned.

In [19], we describe a simple solution for CAN physical layer replication, using off-the-shelf components and yielding enhanced network reliability and availability. The proposed media redundancy mechanisms provide resilience against permanent cabling faults and allow high levels of reliability in the presence of temporary medium faults, effectively enforcing our previous assumptions with regard the failure semantics of CAN network components.

CAN MAC-level properties

CAN has a MAC sub-layer that in essence exhibits the same kind of properties identified in previous works on LANs [22]. A first formalization of CAN MAC-level properties in [20] is complemented in Figure 2 with the definition of the time-related properties MCAN5-MCAN7.

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 multiple factors: traffic patterns, latency classes and offered load bounds, as well as their relation with CAN message identifiers. A number of authors have studied message

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

[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.

[3] Medium Access Control.

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

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. CAN LLC-level properties

schedulability in the CAN fieldbus, using slightly different techniques [21, 25, 11]. The results from these works allow to secure MCAN6.

The *bounded transmission delay* includes T_{ina}, the maximum duration of an inaccessibility fault (MCAN5). During a period of **inaccessibility** a component *temporarily refrains* from providing service on account of a specified transition in its internal state (e.g. network error recovery). 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 [21, 7]. Bounds are established that may be violated upon the occurrence of inaccessibility events. In consequence, the system (e.g. [7]) may exhibit an unpredictable behavior and ultimately fail.

In [18, 24], we have shown this error is bounded and can be known in CAN-based systems. The accommodation of the inaccessibility bounds in the timeliness model (MCAN6) allows to avoid failures due to inaccessibility events.

MCAN7 is crucial for achieving high precision on synchronized clocks [16]. By using CAN controllers with built-in timestamping facilities [12], the value of $\Delta\Gamma_{tight}$ can be bounded by a fraction of the network bit time.

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

Figure 3 recalls from [20] the LLC-level properties of CAN. The first five properties characterize the reliability of CAN communication and its shortcomings. They do dismiss the current belief CAN provides an atomic broadcast service, because LCAN2, LCAN3 and LCAN5 are not in conformity with an atomic broadcast specification [8, 20]. As a matter of fact, CAN does not even guarantee message reliable broadcast [20]. However, property LCAN6 provides the grounds for the design of efficient dependability enforcement mechanisms [20, 17].

4 Reliable Group Communication

One fundamental challenge was then the design of an efficient reliable group communication service for CAN. The current protocol architecture is sketched in Figure 4.

Interfacing the standard CAN layer we use the fundamental fault-tolerant broadcast protocols of [20, 17]: SDCAN enhances LCAN3, by ensuring that each message is delivered **at-most-once**, if no message ordering is required [17]; 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 [20, 17].

A versatile real-time group communication service, offering different *qualities of service*, is defined above this layer.

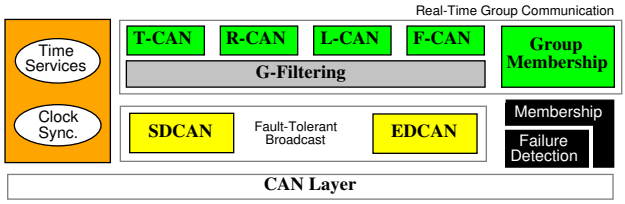

Figure 4. CAN dependable communication layer

The *G-Filtering* sub-layer restricts processing of higher layers to the traffic actually addressed to the node. The top sub-layer includes totally ordered atomic (T-CAN) and reliable (R-CAN) group communication protocols, which are variants of the protocols in [20], and two newly added 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.

5 Failure Detection and Membership

The architecture of Figure 4 includes also the protocols required for node failure detection and membership. A membership service is intended to provide, at any given time, consistent information about failed/correct nodes. A service matching strict application-level latency requirements can be designed with minimal network bandwidth costs (a scarce resource in CAN).

The periodic traffic pattern exhibited by many CAN applications [21, 25] is exploited for that purpose. The periodic high-level messages are implicitly used as heartbeats. 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. A tradeoff exists with a protocol variant where all nodes explicitly issue *life-sign* messages. If a node remains silent during a period longer than the detection latency, that will be a failure. Two optimized micro-protocols ensure consistency of membership information in the presence of: join/leave operations; node failures.

6 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 [16], it is described a clock synchronization algorithm inspired of the generic *a posteriori agreement* algorithm for broadcast networks [23] and of a non fault-tolerant CAN clock synchronization algorithm [5]. Significantly different from those algorithms, the new protocol has been 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 [23] to provide clock synchronization beyond the borders of a single broadcast segment.

7 Towards Object Orientation

Despite the reasonable body of research concerning the use of the object-oriented paradigm in distributed and decentralized systems [2, 6], only a small number of such studies have addressed the particular requirements of embedded systems [1, 4].

Those requirements are even more stringent for distributed embedded systems based on fieldbuses, such as the Controller Area Network. For example, in CAN the network bandwidth does not exceed 1 Mbps and the length of CAN messages data field is limited to eight bytes, which can be insufficient to support distributed object interactions. An identification of the main problems associated with remote object invocation in CAN-based systems and a proposal to achieve real-time guarantees in interactions involving single CAN messages have been discussed in [10].

Relevance of group communication

The availability of group communication is clearly useful in the invocation of replicated objects [10]. Object replication is a common technique to achieve high-level fault-tolerance.

Our reliable group communication layer provides a set of helpful functionalities required to effectively support the invocation of replicated objects in CAN-based systems:

- location transparency in the invocation of replicated objects, through an efficient *group addressing* scheme;
- provision of a versatile service interface, allowing the use of services other than atomic multicast in remote object invocations (e.g. read operations). In particular, the L-CAN protocol exhibits overhead figures only slightly higher than the raw CAN protocol.

- an atomic multicast service designed to exploit the properties of the raw CAN protocol, thus introducing minimum network processing and bandwidth overheads [20];
- transparent support to object interactions requiring data transfers that exceed the maximum CAN message length (e.g. object migration). The F-CAN companion protocol is used in the support of such functionality for any *quality of service*.

Relevance of clock synchronization

The availability of a global notion of time is extremely important for real-time systems. The clock synchronization algorithm of our CAN dependable communications stack allows to establish a global time base with precisions in the order of a few tens of microseconds.

The global timebase can then be used to monitor the temporal behavior of real-time objects. In [10], it is discussed how such functionality can be implemented in CAN-based systems that use the object-oriented paradigm.

Missing services

The reliable group communication and the global time services offered by the CAN dependable communication layer significantly contribute for narrowing the semantic gap between the CAN embedded platform and the higher-level object-orientation layer (Figure 5). However, they do not represent a complete solution.

Given the multi-participant nature of distributed fault-tolerant real-time applications, the object-orientation layer may further benefit from the availability of a set of *group management* functions, identified in [22] and related with the following group activities:

- *competition management* - concerning the interaction of group members with a single entity, which may itself be a group.
- *cooperation management* - concerning inter-group interactions with regard a common purpose.
- *replication management* - that may involve functions such as replication model support (active, semi-active, passive), replica coordination and re-establishment of replication level upon replica failure [15].

These functions exhibit very different degrees of complexity. While *competition management* requirements may be satisfied almost entirely by the group communication services alone, *replication management* requires rather complex software components. The efficient implementation of group management functions in CAN will be pursuit in our future research.

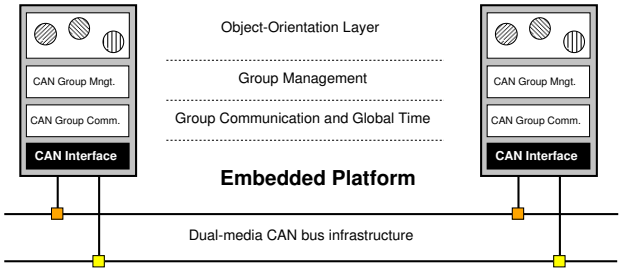

Figure 5. Narrowing the gap between object-orientation layer and the CAN embedded infrastructure

8 Concluding Remarks

The combination of object-orientation with distribution, fault-tolerance and real-time involves the resolution of non-trivial problems, particularly when dealing with embedded systems. However, the large semantic gap between a high-level object-orientation layer and the embedded infrastructure can be narrowed, should the embedded system platform provide semantically rich services such as those offered by group communication, failure detection and membership, clock synchronization and group management.

In this paper we have provided an overview of our research effort for providing those services in the context of CAN, the Controller Area Network.

Reasoning about CAN communications, we have identified infrequent but plausible fault scenarios that prevent CAN of providing reliable or atomic broadcast. In consequence, we formalized the properties actually secured by CAN in a systemic system model and used those results as a basis for constructing a suite of protocols for dependable real-time communication.

References

[1] L. Bacellar and B. Upender. A dependable distribution-transparent remote method invocation model for object-oriented distributed embedded computer systems. In *Proceedings of the 1st International Symposium on Object-Oriented Real-Time Distributed Computing*, Kyoto, Japan, Apr. 1998. IEEE.

[2] J. Briot, R. Guerraoui, and K. Lohr. Concurrency and distribution in object-oriented programming. *ACM Computing Surveys*, 30(3):291–329, Sept. 1998.

[3] CiA - CAN in Automation. *CAN Physical Layer for Industrial Applications - CiA/DS102-1*, Oct. 1992.

[4] M. Gergeleit, J. Kaiser, and H. Streich. DIRECT: Towards a distributed object-oriented real-time control system. In *Proceedings of the Workshop On Concurrent Object-Based Systems*, Dallas, Texas, USA, Oct. 1994.

[5] 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, Sept. 1994. CiA.

[6] A. Gheith and K. Schwan. CHAOS: Kernel support for multiweight objects, invocations, and atomicity in real-time multiprocessor applications. *ACM Transactions on Computer Systems*, 11(1):33–72, Feb. 1993.

[7] 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.

[8] V. Hadzilacos and S. Toueg. Fault-tolerant broadcasts and related problems. In S. Mullender, editor, *Distributed Systems*, ACM-Press, chapter 5, pages 97–145. Addison-Wesley, 2nd edition, 1993.

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

[10] J. Kaiser and M. Livani. Invocation of real-time objects in a can bus-system. In *Proceedings of the 1st International Symposium on Object-oriented Real-Time distributed Computing*, Kyoto, Japan, Apr. 1998. IEEE.

[11] M. Livani, J. Kaiser, and W. 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.

[12] Motorola, Inc., USA. *MPC555 User's Manual*, May 1998.

[13] Red-can a fully redundant can-system. NOB Elektronik AB Product Note - Sweden. http://www.nob.se.

[14] Philips Semiconductors. *TJA1053 - Fault-tolerant CAN transceiver*, Oct. 1997.

[15] D. Powell, editor. *Delta-4 - A Generic Architecture for Dependable Distributed Computing*. ESPRIT Research Reports. Springer Verlag, Nov. 1991.

[16] L. Rodrigues, M. Guimarães, and J. Rufino. Fault-tolerant clock syncronization in CAN. In *Proceedings of the 19th Real-Time Systems Symposium*, pages 420–429, Madrid, Spain, Dec. 1998. IEEE.

[17] 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, Sept. 1998. IEEE.

[18] 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, Oct. 1995. CiA.

[19] 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*, Madison, Wisconsin - USA, June 1999. IEEE. (to appear).

[20] 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.

[21] 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, Sept. 1994. CiA.

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

[23] 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.

[24] 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.

[25] 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, Apr. 1997.