Enforcing Dependability and Timeliness in Controller Area Networks

José Rufino
FCUL
Campo Grande
Lisboa, Portugal
ruf@di.fc.ul.pt

Carlos Almeida
IST/UTL
Av. Rovisco Pais
Lisboa, Portugal
cra@comp.ist.utl.pt

Paulo Veríssimo
FCUL
Campo Grande
Lisboa, Portugal
pjv@di.fc.ul.pt

Guilherme Arroz
IST/UTL
Tagus Park
Oeiras, Portugal
pcegsa@alfa.ist.utl.pt

Abstract—**Standard fieldbuses, such as the Controller Area Network (CAN), are today a cost-effective solution for distributed computer control systems. However, the standard CAN protocol exhibits a set of severe shortcomings in respect to the provision of strict dependability and timeliness guarantees. This paper identifies those shortcomings and discusses the main design challenges we have been tackling in a comprehensive way to provide a CAN-based infrastructure support for extremely reliable hard real-time communications, dubbed CAN Enhanced Layer (CANELy).**

I. 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.

The Controller Area Network (CAN) is traditionally viewed as a very robust fieldbus infrastructure. However, we have identified a set of limitations of the standard CAN protocol with regard to the provision of strict availability, reliability and timeliness guarantees. Thus, one crucial question is: would the lack of those attributes be due to a fundamental inability, in which case there is little chance that any hardware or software will solve the problem, or else, would it be due to some insufficiency in functionality, in which case it can be mended by adding the latter?

We have realized that what was missing in the native CAN fieldbus to attain high levels of dependability was indeed a set of fault tolerance and timeliness-related services. Moreover, we have shown that these can be provided off-the-shelf (i.e. without modifications to the CAN standard or to existing CAN controllers), through the use of properly encapsulated additional software/hardware components.

We call the materialization of this concept **CAN Enhanced Layer** (CANELy), which is made from several hardware/software building blocks [22], [17], [19], [20], [21]. This paper provides an overview of the main design issues to be addressed to enforce dependability and timeliness in the CANELy architecture.

Faculdade de Ciências da Universidade de Lisboa, Campo Grande - Bloco C8, 1749-016 Lisboa, Portugal. Tel: +351-217500254, Fax: +351-217500084. Navigators Home Page: http://www.navigators.di.fc.ul.pt.

This work was partially supported by FCT through Project POSC/EIA/56041/2004 (DARIO).

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.

The paper is organized as follows: Section II provides a short description of CAN and analyzes its dependability; Section III discusses the system model; Section IV analyzes how to improve the availability of the network infrastructure; Section V discusses how to secure CAN timely behavior in the presence of faults and Section VI presents a CAN-oriented protocol suite, which includes reliable group communication, failure detection and membership, and clock synchronization services; Section VII concludes the paper.

II. Controller Area Network

CAN is a multi-master fieldbus that uses a twisted pair cable as transmission medium [7], [4]. 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*, also the state of an idle bus; *dominant*, which always overwrites a recessive value. This behavior, together with the use of unique 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 disseminated on the network. It may contain a *message*, a user-level piece of information.

In the signaling of abnormal network operation incidents, the CAN protocol uses: *error frames*, for (global) error signaling; *overload frames*, to react to violations of the standard interframe spacing [7].

Basic dependability of CAN

Though CAN fault-confinement and error detection mechanisms ensure that most failures are perceived consistently by all nodes, some subtle errors can lead to inconsistency and induce the failure of dependable communication protocols based on CAN operation alone [22].

Inconsistent frame omissions may occur when faults hit the last two bits of a frame at some nodes, tagged $\times$ set[1] in Figure 1-B, which may cause: the message to be accepted in duplicate by a subset of recipients, upon retransmission;

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

inconsistent message omission, if the sender fails before retransmission. However infrequent these failure scenarios may be, the probability of its occurrence is high enough to be taken into account for highly fault-tolerant hard real-time applications of CAN [22], [5].

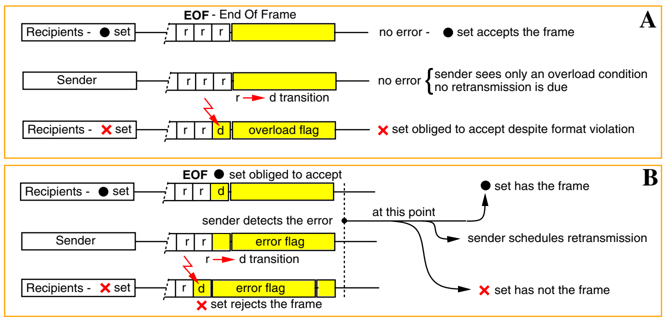

Fig. 1. Inconsistency in CAN error handling

In addition, the occurrence of certain incidents in CAN operation (such as: bit errors; transmitter/receiver glitches) produces a subtle form of (virtual) network partitioning, that we call *inaccessibility*. Though the standard CAN protocol has means of recovering from these situations, the recovery process takes time, leading to increase the network access delay as seen by one or more nodes. This may induce failures of expected hard real-time properties of the network. Timeliness of applications may be at stake, which is non acceptable in dependable hard real-time systems. Thus, provisions to tolerate such kind of (inaccessibility) faults are required [27], [28].

Other problem concerns the availability of the network infrastructure. CAN is traditionally viewed as a robust fieldbus. The physical layer specified in [7] 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 [13]. 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 [12], [19].

III. System Model

The definition of a systemic model for CAN proved extremely useful: not only did it show the weaknesses of CAN with regard to dependability, but it provided the grounds to handle those problems effectively. Next, we enumerate our fault assumptions for the system and discuss a relevant set of CAN protocol properties, drawn from our previous works on CAN [22], [19], [18].

Fault model

Let us assume a CAN infrastructure composed of $\mathcal{N}$ nodes interconnected by a Channel. The Channel is the physical path – cable medium and transceivers – used by the MAC[2] entities to communicate.

[2]Medium Access Control.

We define: a component is **weak-fail-silent** if it behaves correctly or crashes if it exhibits more than a given number of omission failures, called the component's *omission degree* [27], in a time interval of reference. In respect to CAN components, an omission is an error that destroys a data or remote frame. Thus, the following failure semantics are defined for the **CAN network components**:

- individual components are **weak-fail-silent** with *omission degree* f_o;
- failure bursts never affect more than f_o transmissions in a time interval of reference[3];
- omission failures may be inconsistent (i.e., not observed by all recipients);
- there is no permanent failure of the Channel (e.g. the simultaneous partitioning of all redundant media [19]).

The omission degree is a general measure of the reliability of the CAN components to transient errors: failure bursts affect at most f_o transmissions in the time interval of reference. However, for the particular set of channel redundant media, it is made the additional assumption that failures in different media are independent.

CAN physical-level properties

The foundation of CAN operation is described by the set of physical layer properties formalized in Figure 2.

PCAN1 - *Bit* **Simultaneity**: for any *Bit* p of any transmitter s starting at $t^s_B(p)$, if $t^r_B(p)$ is the start of *Bit* p as seen by receiver r, for any r, then in absence of faults, $t^s_B(p) = t^r_B(p)$.

PCAN2 - Wired-AND Multiple Access: for all transmitters s in $\mathcal{N}$, the value of any *Bit* p seen by the channel c is, in absence of faults, $v^c_B(p) = \prod_{s \in \mathcal{N}} v^s_B(p)$.

PCAN3 - *Bit* **Broadcast**: in absence of faults, for any *Bit* p on the channel c, and for any receiver r, $v^r_B(p) = v^c_B(p)$.

Fig. 2. CAN physical-level properties

Property PCAN1 formalizes the *quasi-stationary* propagation of signals in the CAN Channel [24], [19]. A *Bit* is a time interval of constant nominal duration, being $t^s_B(p)$ the (unobservable) real time instant when *Bit* p starts at s (s is a transmitter, a receiver or the channel). A single *Bit* is broadcast on the channel at a time, as described by properties PCAN1 and PCAN3. In absence of faults, a *Bit* p at s assumes one and only one logical value $v^s_B(p)$. The symbol $\prod$ is used in PCAN2 to specify a logical AND function combining the signals from multiple simultaneous transmitters into a single *Bit* value. This is in conformity with standard CAN implementations [7], [4], [13].

Properties PCAN1 to PCAN3 are required by the CAN protocol for arbitrating accesses to the shared medium, bus state monitoring and data transfer. A thorough understanding of CAN physical layer properties is of utmost importance in the definition of dependability enforcement mechanisms, such as the design of an innovative method to implement bus-based media redundancy in CAN [19].

[3]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.

CAN MAC-level properties

The CAN protocol has a MAC sub-layer that exhibits the same kind of properties identified in LANs [27]. Figure 3 enumerates a relevant set of CAN MAC-level properties.

MCAN1 - Error Detection: correct nodes detect any corruption done by the network in a locally received frame.

MCAN2 - Bounded Omission Degree: in a known time interval T_{rd}, omission failures may occur in at most k transmissions.

MCAN3 - 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}.

MCAN4 - Bounded Transmission Delay: any frame queued for transmission is transmitted on the network within a bounded delay of $T_{td} + T_{ina}$.

Fig. 3. CAN MAC-level properties

Property MCAN1 derives directly from CAN built-in error handling mechanisms. It implies that frame errors are transformed into omissions [18]. The residual probability of undetected frame errors is negligible [3]. Property MCAN2 maps the failure semantics of our model onto CAN operational assumptions, being $k \geq f_o$. This is crucial to achieve reliable hard real-time operation on CAN.

The behavior of CAN in the time-domain is described by property MCAN4, which specifies a maximum frame transmission delay. In the absence of faults, T_{td} includes the normal queuing, access and transmission delays, and depends on message latency classes and offered load bounds [25], [30], [10]. In general, T_{td} also includes the extra delays resulting from the additional queuing effects caused by the *periods of inaccessibility* [14], [15], [2]. The bounded frame transmission delay includes T_{ina}, a corrective term that accounts for the worst-case duration of inaccessibility events, given the bounds specified by property MCAN3. This way, timing failures due to inaccessibility incidents can be avoided [21]. The inaccessibility characteristics of CAN are obtained by analysis of the CAN protocol [28].

CAN LLC-level properties

The standard CAN LLC[4] sub-layer, built on top of the basic MAC functionality, has error-recovery mechanisms yielding interesting message-level properties. While the omission failures specified by MCAN2 are masked in general at the LLC-level by the retry mechanism of CAN [7], the existence of inconsistent omissions (cf. §II) implies that: there may be message duplicates when they are recovered; that some j of the k omissions will show at the LLC interface as inconsistent omissions.

Figure 4 summarizes the LLC-level properties of CAN. Properties LCAN1 and LCAN2 characterize the shortcomings of CAN with regard reliability. They do dismiss the belief CAN provides an atomic broadcast service, because LCAN1 and LCAN2 are not in conformity with an atomic broadcast specification [6], [22]. In fact, CAN does not even guarantee message reliable broadcast [22].

[4]Logical Link Control.

LCAN1 - 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.

LCAN2 - At-least-once Delivery: any message delivered to a correct node is delivered at least once.

LCAN3 - Bounded Inconsistent Omission Degree: in a known time interval T_{rd}, inconsistent omission failures may occur in at most j transmissions.

Fig. 4. CAN LLC-level properties

Property LCAN3 is thus of fundamental importance, since it provides the grounds to efficiently enforce reliability attributes in CAN communications [22], [17], [20].

IV. Network Availability

The first problem we address concerns the availability of the CAN infrastructure. An existing commercial solution [12] does not solve the problem efficiently: it is implemented as a self-healing ring/bus architecture; ring reconfiguration takes time (it can last as long as 100 ms) and meanwhile the network is partitioned.

To enhance network availability one rely on the replication of the physical path (bus medium and transceivers) used by 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.

Basic Mechanisms

An ingenious solution to handle replicated media has been originally introduced in [19]. The wired-AND nature of CAN, as described by property PCAN2 (cf. Figure 2), is extended to the media interface level: the signals from the different redundant media receivers are combined in a conventional AND function, before interfacing the standard MAC sub-layer. This simple method, feasible given property PCAN1, ensures resilience to medium physical partitions and stuck-at-recessive failures [19].

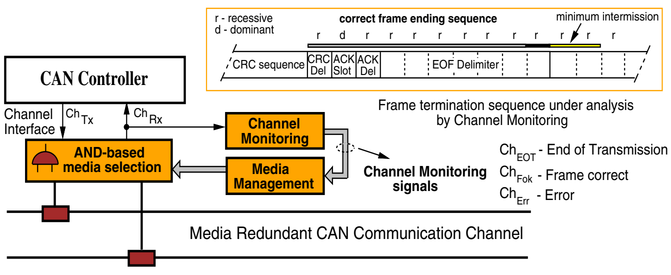
r - recessive
d - dominant
correct frame ending sequence
minimum intermission
r d r r r r r r r r r r r
CRC sequence
CRC Del
ACK Slot
ACK Del
EOF Delimiter
CAN Controller
Channel Interface
Ch_{Tx}
Ch_{Rx}
Frame termination sequence under analysis by Channel Monitoring
Channel Monitoring
AND-based media selection
Media Management
Channel Monitoring signals
Ch_{EOT} - End of Transmission
Ch_{Fok} - Frame correct
Ch_{Err} - Error
Media Redundant CAN Communication Channel

Fig. 5. CAN-oriented Media Redundancy Mechanisms

The **AND-based Media Selection** module of Figure 5 also includes the resources required to selectively disable/enable each medium interface, e.g. upon failure/after repair. The other modules depicted in Figure 5 provide additional monitoring and fault treatment functions.

The **Channel Monitoring** module provides a basic set of signals (cf. Figure 5), given the observable behavior of CAN at the PHY[5]-MAC interface [19].

The **Media Management** module uses those signals, together with media monitoring functions/signals (not shown in Figure 5, for the sake of simplicity) to:

- perform receiver-based frame monitoring, comparing Channel and Medium incoming frame data on a bit-by-bit basis;
- detect and account for omissions at each Medium interface;
- disable Medium operation, until repair, if it exceeds the allowed omission degree bound or if a stuck-at-dominant failure is detected.

These mechanisms are not hard to implement and provide effective resilience against *all* the cabling failures discussed in Section II.

Enhanced Dependability Mechanisms

Additional dependability enforcement measures specified in [18] are currently being integrated with advanced mechanisms in the design of an enhanced Media Management module, allowing:

- detection of medium partition and medium stuck-at-recessive failures. Though such kind of failures are tolerated by the bare AND-based media selection, its detection and signaling to high-level (user) management entities do allow repair actions essential to the preservation of dependability coverage [18];
- early detection of stuck-at-dominant Channel failures, allowing a prompt shutdown of the incorrect node [18];
- provision of advanced CAN-oriented *media quarantine* techniques, where "faulty" media are temporarily disabled, to allow operation to "proceed" with the "correct" set. A medium is disabled as soon as a bit mismatch is detected on Medium and Channel interfaces. This technique, dubbed as *early quarantine*, allows to achieve the ambitious goal of having a Channel with an omission degree bound $k=1$ (cf. property MCAN2, in Figure 3), meaning no more than one consecutive frame omission occurs in the Channel.

Using Full Network Redundancy

The last issue to be discussed in respect to the availability of the CAN infrastructure concerns the option, specified in [18], of combining simple media and full network redundancy in the same infrastructure (cf. Figure 6). In CAN this can be achieved with small overhead costs: each network cable may include an additional differential pair, to support a dual-MAC interface; many low-cost controllers already include a second CAN interface[6].

Moreover, the architecture of Figure 6 can be made compliant with the standard physical layer specification, by using spare pins in the CiA connector [4].

The dual-bus attachment can be used to improve the overall system dependability, contributing to secure glitch-free communication and tight hard real-time message delivery guarantees, thus opening room for the use of CAN in safety-critical applications.

[5]PHY, stands for the Physical layer.

[6]For example, the state-of-the-art Maxim/Dallas Semiconductors High-Speed Microcontroller DS80C390 integrates in a single chip a 80C52 processor core and two advanced CAN 2.0B controllers [11].

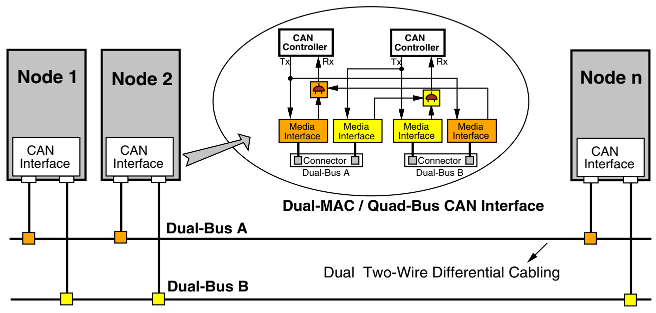

Fig. 6. Highly-available dual-bus media redundant infrastructure

V. Control of Inaccessibility

The normal operation of LANs and fieldbuses, CAN included, can be hindered by *periods of inaccessibility*, which derive from incidents in network operation (e.g. bit errors, transmitter/receiver glitches) that temporarily prevent communication. Service is not provided to some or all of the nodes and this may have the effect of increasing the corresponding network access delay.

Analysis of message transmission latencies performed under the assumption the network always operates normally [25], [30], [10] are relevant and, undoubtedly, useful for optimal system configuration. However, bounds are established that may be violated upon the (even if rare) occurrence of inaccessibility events.

To avoid timing failures due to network inaccessibility incidents [28], [18], [21], it is required to:

- **I1 -** study the accessibility constraints, ensuring that the number of inaccessibility periods and their duration have a bound (MCAN3), as done in [28];
- **I2 -** show that such a bound is suitably low for service requirements, as specified in [27];
- **I3 -** accommodate the effects of inaccessibility events in the timeliness model and in protocol operation, at all the relevant levels of the system [18], [21].

The control of inaccessibility is crucial in simplex networks, even if redundant media is being used. In addition, it cannot be ignored with full network redundancy, since inaccessibility affecting individual network replicas would lower their fault coverage, that is, the probability that each of them is correct (in the time domain). In this sense, the mechanisms for the control of inaccessibility are also applicable to individual replicas of a redundant network.

CAN Inaccessibility Boundedness

In respect to point I1, the analysis in [28], [18] provides a comprehensive set of easy-to-use formulas to evaluate the worst-case bounds of the periods of inaccessibility.

The results of such analysis are summarized in Figure 7. The dependability enforcement mechanisms introduced in Section IV has induced a reduction of inaccessibility worst-case bounds for some scenarios, as shown in Figure 7. It is worth noticing: single bit errors (on the leftmost part of Figure 7) are not reduced because they affect only the transmission of one frame; the worst-case inaccessibility bound for bus multi-burst errors is reduced, due to the effects of the *early quarantine* mechanisms discussed in §IV;

the low worst-case figure of the bus reconfiguration delay (209 μs @ 1Mbps), compared with other failure scenarios and, in particular, with the 100 ms of commercial systems currently available [12].

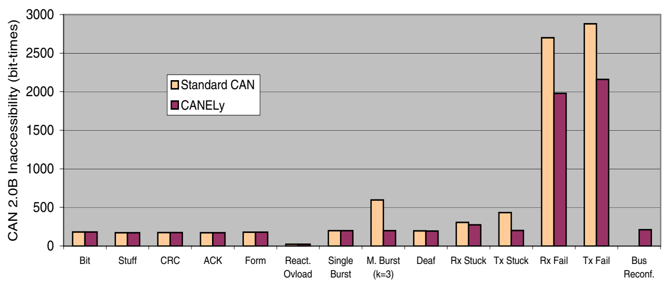

Fig. 7. Normalized durations of inaccessibility incidents

On the other hand, the actions taken in [18] to enforce the weak-fail-silent assumption for the network components: are based on CAN own error confinement mechanisms [22]; induced only a moderate, though interesting, reduction of inaccessibility durations for receiver and transmitter failure scenarios, as shown in Figure 7.

The avoidance of "babbling idiot" failures has further been studied: the inaccessibility constraint derived in [1] for CAN settings has a normalized duration of 41 bit-times, much lower than the values inscribed in Figure 7; thus, it does not represent a worst-case bound. Babbling idiot failures are not detectable by the native CAN error handling mechanisms. Protection has to be provided by special-purpose machinery (bus guardian) [26], [1].

CAN Message Schedulability Analysis

Securing condition I2 requires a comprehensive analysis of message schedulability guarantees given known traffic patterns and offered load bounds. Both error free and worst-case error analysis are relevant. The former, is intended to provide the parameters required for optimal system configuration [25], [30], [10]. The latter, given a worst-case pattern of inaccessibility incidents, provides hard real-time guarantees of message schedulability and defines worst-case message delivery delays [14], [15], [2].

Extended versions of existing message schedulability analysis tools and methodologies [14], [25], [16], [23] should be able to provide a set of relevant parameters for worst-case system configuration, including a bound for the time the effects of inaccessibility last in the system.

CAN Inaccessibility Control

Finally, we need to address point I3. In [21] we have thoroughly discussed how the error handling functions required for media redundancy could be extended to include the functionality needed for:

- the provision of an indication of occurrence of an inaccessibility incident, to be active as soon as inaccessibility is detected and for how long its effects last in the message queues;
- the evaluation of the real durations of inaccessibility incidents and of the extra message queuing and network access delays;
- the evaluation of inaccessibility upper bounds with respect to the total number of incidents, i, and their total duration, T_{ina}, in a period of reference (cf. property MCAN3);
- the evaluation of the worst-case duration of the entire period where the effects of inaccessibility last in the system, which we have defined as inaccessibility epoch, T_{p_ina}.

These mechanisms are not hard to implement, since they can be engineered as simple low-cost extensions to the bus media redundancy machinery, described in §IV. Both mechanisms can be easily integrated in a single, medium capacity, programmable logic device [29].

The effects of inaccessibility must now be included in protocol execution and in the timeliness calculations, both at the application level and at the low-level protocols.

At application-level, the control of inaccessibility effects is simple: a corrective term accounting for the worst-case duration of an inaccessibility epoch is transparently added to optimal timeout values [21]. At the low-level protocols, advanced inaccessibility control mechanisms [21] allow: to account for the real duration of an inaccessibility epoch; to selectively add a corrective term to (optimal) timeout values, only when inaccessibility affects protocol timeliness.

The mechanisms introduced for the control of inaccessibility also allow the assessment of real system parameters (w.r.t. timing, omission), making possible to monitor the coverage of both dependability and timeliness models.

VI. Reliable Communications

The last issue to be discussed concerns the design of an efficient reliable group communication service for CAN, taking into account the shortcomings of the CAN standard layer with regard to dependability (cf. §II).

A modular approach has been followed in the design of the architecture sketched in Figure 8. Interfacing the CAN standard layer, a fundamental set of fault-tolerant broadcast protocols [22]: enhance LCAN2, ensuring that each message is delivered **at-most-once**; enhance LCAN1, removing the condition of the sender not failing; introduces order attributes, thus securing all the properties of an atomic broadcast service [6], [22].

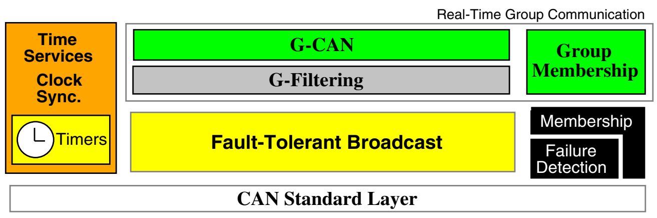

Fig. 8. Reliable group communication protocol suite

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 actually addressed to the node. On top, *G-CAN* includes CAN-oriented totally ordered atomic and reliable group communication protocols. A companion protocol, in *G-CAN*, exploits CAN properties to support an efficient message fragmentation scheme that does not need to use sequence numbers for fragment ordering.

The reliable communications protocol suite also includes provisions for: node failure detection[20]; node and group membership; time and clock synchronizations services[17].

VII. Concluding Remarks

Given the increasing demand for distributed fault-tolerant systems based on fieldbus technologies, we have: investigated the shortcomings of CAN, with regard to dependability and timeliness; defined a systemic model of CAN that not only did it show those weaknesses, but it provided the grounds to handle those problems effectively.

This paper has discussed the main components in the CANELy architecture, the CAN Enhanced Layer [18], a combination of the CAN standard layer with some simple machinery resources and low-level protocols achieving highly dependable real-time communications, described in several publications [28], [22], [17], [19], [20], [21].

The main attributes of CANELy and their comparison both with the stand-alone CAN and with the industry standard Time-Triggered Protocol (TTP) [9], [8], are outlined in Figure 9. These results are a contribution to dismiss ideas that CAN is not suited for designing hard real-time systems with very high dependability requirements.

Parameter	TTP	CAN	CANELy
Communications	broadcast	broadcast	multicast/ broadcast
Omission handling	masking	detection/ recovery	both algorithms
	diffusion	retransmission	
Inaccessibility duration	unknown	14 - 2880 *bit-times*	14 - 2160 *bit-times*
Inaccessibility control	not completely addressed	no	application and low-level
Media redund.	no	no	yes
Channel redund.	yes	no	yes (optional)
Babbling idiot avoidance	bus guardian	not provided	can be added (cf. [1])
Resilience to lack of coverage	never-give-up strategy	none	detects violation of bounds

Fig. 9. Comparison of dependability and timeliness-related attributes of TTP, Standard CAN and CAN Enhanced Layer (CANELy)

References

[1] I. Broster and A. Burns. An analysable bus-guardian for event-triggered communication. In *Proc. of 24th Real-time Systems Symposium*, Cancun, Mexico, December 2003. IEEE.

[2] I. Broster, A. Burns, and G. Rodríguez-Navas. Probabilistic analysis of CAN with faults. In *In Proc. of 23rd Real-time Systems Symposium*, Austin, Texas, December 2002. IEEE.

[3] J. Charzinski. Performance of the error detection mechanisms in CAN. In *Proceedings of the 1st International CAN Conference*, pages 1.20–1.29, Mainz, Germany, September 1994. CiA.

[4] CiA - CAN in Automation. *CAN Physical Layer for Industrial Applications - CiA Draft Standard 102 Version 2.0*, April 1994.

[5] J. Ferreira, A. Oliveira, P. Fonseca, and J. Fonseca. An experiment to assess bit error rate in CAN. In *Proc. of the 3rd. Int. Workshop on Real-Time Networks*, Catania, Italy, June 2004.

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

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

[8] H. Kopetz. A comparison of CAN and TTP. In *Proceedings of the 15th IFAC Workshop on Distributed Computer Control Systems*, Como, Italy, September 1998. IFAC.

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

[10] 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.

[11] Maxim/Dallas Semiconductors. *DS80C390 Dual-CAN High-Speed Microprocessor*, February 2005.

[12] RED-CAN a fully redundant CAN-system. NOB Elektronik AB Product Note - Sweden. http://www.nob.se.

[13] Philips Semiconductors. *TJA1053 - Fault-tolerant CAN transceiver*, October 1997.

[14] L. Pinho, F. Vasques, and E. Tovar. Integrating inaccessibility in response time analysis of CAN networks. In *Proceedings of the 3rd International Workshop on Factory Communication Systems*, pages 77–84, Porto, Portugal, September 2000. IEEE.

[15] S. Punnekkat, H. Hansson, and C. Norstrom. Response time analysis under errors for CAN. In *Proceedings of the Real-Time Technology and Applications Symposium*, pages 258–265, Washington, USA, May 2000. IEEE.

[16] J. Rodrigues, J. Ventura, and L. Rodrigues. Schedulability analysis of an event-based real-time protocol framework. In *Proceedings of the Seventh International Workshop on Object-Oriented Real-Time Dependable Systems*, pages 1530–1443, San Diego, USA, January 2002. IEEE.

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

[18] J. Rufino. *Computational System for Real-Time Distributed Control.* PhD thesis, Technical University of Lisbon - Instituto Superior Técnico, Lisboa, Portugal, July 2002.

[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.

[20] J. Rufino, P. Veríssimo, and G. Arroz. Node failure detection and membership in CANELy. In *Proceedings of the 2003 International Conference on Dependable Systems and Networks*, pages 331–340, San Francisco, California, USA, June 2003. IEEE.

[21] J. Rufino, P. Veríssimo, G. Arroz, and C. Almeida. Control of inaccessibility in CANELy. In *Proceedings of the 6th. International Workshop on Factory Communication Systems*, pages 35–44, Torino, Italy, June 2006. IEEE.

[22] 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.

[23] M. Santos, M. Stemmer, and F. Vasques. Schedulability analysis of messages in a CAN network applied to an unmanned airship. In *Proceedings of the 28th Annual Conference of the IEEE Industrial Electronics Society*, volume 3, pages 1909–1914, Sevilla, Spain, November 2002. IEEE.

[24] R. Stuart. CAN bit timing requirements. Application Note AN1798, Motorola, Inc., East Kilbride, Scotland, 1999.

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

[26] K. Tindell and H. Hansson. Babbling idiots, the dual-priority protocol and smart CAN controllers. In *Proceedings of the 2nd International CAN Conference*, pages 7.22–7.28, London, England, October 1995. CiA.

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

[28] 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.

[29] Xilinx. *Spartan and Spartan-XL Families Field Programmable Gate Arrays Data Sheet*, June 2002.

[30] 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.