Supporting Networked Control Systems: Assessment of the CAN Protocol Considering Periods of Network Inaccessibility

Luís Miguel Pinho [1], Francisco Vasques [2]

[1] Department of Computer Engineering, ISEP, Polytechnic Institute of Porto
Rua São Tomé, 4200 Porto, Portugal

[2] Department of Mechanical Engineering, FEUP, University of Porto
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

e-mail: lpinho@dei.isep.ipp.pt, vasques@fe.up.pt

Abstract. *Due to the increased availability of low cost network technology, the use of networks to interconnect sensors, actuators and controllers is becoming widely accepted for the implementation of feedback control systems. Such type of feedback implementation, wherein the control loops are closed through a real-time network, are called Network Controlled Systems (NCS).*

When implementing a NCS, the communication network must provide a timely communication service to the control application. Nevertheless, it must be understood that the continuity of service is not fully guaranteed, since it may be disturbed by temporary periods of network inaccessibility. Therefore, the assessment of the network responsiveness considering such inaccessibility periods is a fundamental issue. In this paper we integrate state-of-the-art inaccessibility studies with the response time analysis of CAN networks, providing an accurate analysis of its responsiveness.

1 Introduction

Fieldbus networks are becoming increasingly popular in computer-controlled systems. Fieldbus allow field devices like sensors, actuators and controllers to be interconnected at low cost, using less wiring and requiring less maintenance than point-to-point connections. Besides the economical aspects, the use of Fieldbus in computer-controlled systems is also reinforced by the increasing decentralization of control and measurement tasks. Computer-controlled systems wherein the control loops are closed through a real-time fieldbus network are called Network Controlled Systems (NCS) (Figure 1).

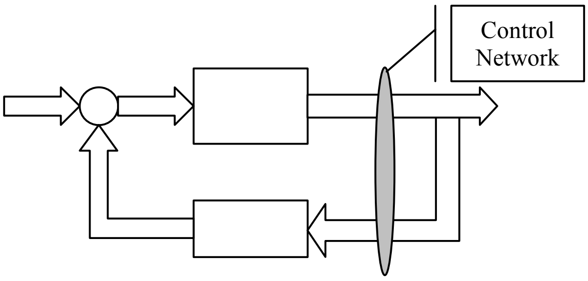

Figure 1: Example of a Network Controlled System (NCS)

Using a control network to interconnect sensors, actuators and controllers in a feedback control system, requires the use of a control network that must be simultaneously:

a) able to support periodic message streams, in order to convey the control-related periodic data between the controller and the set of related sensors / actuators;
b) able to guarantee upper-bounded response times for the message transfers, in order to cope with the control-related delays;
c) able to guarantee a predictable timing behavior in the presence of a variable network load due to traffic non related to the control application (such as: alarms, surveillance video streams, etc.);
d) and, above all, able to guarantee a predictable timing behavior in the presence of a faulty communication behavior.

In addition, a well-known problem when using a control network is the presence of induced jitter, that is, the variability of the time interval between consecutive transfers. For instance, in spite of periodically requesting the transfer of a specific sensor value, the actual transfer will not be immediately executed, as messages need to be scheduled for transmission in a shared resource (the communication medium). As a consequence, in some cycles the sensor message will be transferred earlier in the cycle period, and in some other cycles it will be transferred later. The real-time service provided by the control network will just guarantee that the sensor message will always be transferred before its deadline.

The assessment of the control network responsiveness must be focused on the analysis of the above-mentioned real-time properties of the communication protocol, in the presence of a faulty communication behavior.

Controller Area Network (CAN) [1] was originally designed for use within road vehicles, to solve cabling problems arising from the growing use of microprocessor-based components in vehicles. Due to its very interesting characteristics, CAN is also being considered for the automated manufacturing and distributed process control environments [2], which target small-scale Network Controlled Systems (NCS). Several studies on how to guarantee the timing requirements of messages in CAN networks are available (e.g. [3]), thus providing pre-run-time schedulability conditions for the analysis of the timing requirements of NCS traffic.

One of the perceived drawbacks of communication networks is that continuity of service is not fully guaranteed, since it may be disturbed by temporary periods of network inaccessibility (periods during which stations cannot communicate with each other, due to the existence of on-going error detection and recovery mechanisms). A study on the inaccessibility characteristics of CAN networks has been presented at [4], identifying the duration of its error detection and recovery periods.

This paper addresses the integration of such inaccessibility studies with the response time analysis of CAN fieldbus networks, providing a more accurate analysis of its real-time behaviour. Essentially, formulae are provided to evaluate both the response time of messages and the resulting network load, considering a realistic behaviour, where the CAN network is disturbed by periods of inaccessibility.

The remainder of this paper is organised as follows. Section 2 describes the most important characteristics of CAN networks. Particular relevance is given to its error detection and recovery mechanisms. Section 3 describes some of the most relevant existent works on response time analysis and inaccessibility studies of CAN networks.

Based on the characteristics of the CAN protocol, in Section 4 the response time analysis of CAN networks is extended, to consider both the case of Remote Frames and to integrate inaccessibility issues. In that Section we derive both response time and network load analyses for CAN networks. We consider a realistic assumption, where the CAN communication network is disturbed by periods of inaccessibility. A benchmark is used to compare the proposed analysis with the classical response time analysis of CAN messages. The achieved results emphasise that, in the presence of bus errors, a CAN fieldbus network is not able to provide different integrity levels, since errors in low priority messages interfere with the response time of higher priority messages. An important conclusion is also that, even for reduced network loads, the system may become unschedulable in the presence of station errors. Finally, in Section 5, we analyse the pessimism inherent to the proposed analysis.

2 A Brief Description of the CAN Protocol

2.1 Main Characteristics of the CAN Protocol

The CAN protocol implements a priority-based bus, with a carrier sense multiple access with collision avoidance (CSMA/CA) MAC. In this protocol, any station can access the bus when it becomes idle. However, contrarily to Ethernet-like networks, the collision resolution is non-destructive, in the sense that one of the messages being transmitted will succeed.

There are 4 types of frames that can be transferred in a CAN network. Two of them are used during the normal operation of the CAN network: the Data Frame, which is used to transfer data from one station to another and the Remote Frame, which is used to request data from a distant station. The other two frames are used to signal an abnormal state of the CAN network: the Error Frame signals the existence of an error state and the Overload Frame signals that a particular station is still not ready to transmit data.

Figure 2 shows the structure of a Data Frame. Description for the specific fields (SOF, Identifier, RTR, IDE, r0, DLC, CRC and EOF) can be found in [1]. A Remote Frame has the same structure of a Data Frame (without data field) with the same identifier of the remotely requested Data Frame. The structure of both the Error and Overload Frames will be presented in sub-section 2.2.

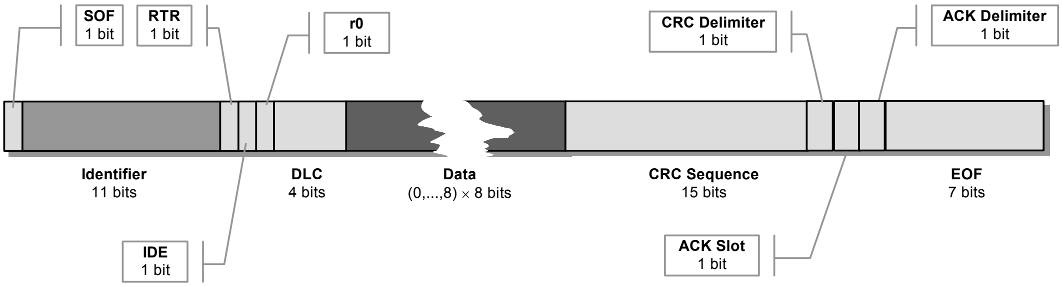

Figure 2 - Structure of a CAN Data Frame

Bus signals can take two different states: *recessive bits* (idle bus), and *dominant bits* (which always overwrite recessive bits). The collision resolution mechanism works as follows: when the bus becomes idle, every station with pending messages will start to transmit. During the transmission of the identifier field, if a station transmitting a recessive bit reads a dominant one, it means that there was a collision with at least one higher-priority message, and consequently this station aborts the message transmission. The highest-priority message being transmitted will proceed without perceiving any collision, and thus will be successfully transmitted. The highest priority message is the one with most leading dominant bits on the identifier field. Obviously, each message stream must be uniquely identified. The station that lost the arbitration phase will automatically retry the transmission of its message.

2.2 Error Detection and Recovery Mechanisms on the CAN Protocol

In the CAN protocol, all the stations continuously monitor every frame being transmitted on the bus, to detect any transmission error. The station which firstly detects an error, starts the transmission of an Error Frame (which starts with 6 consecutive dominant bits). The transmission of an Error Frame is an efficient way for the CAN protocol to tolerate transient failures (e.g. due to electromagnetic interference).

This Error Frame transmission is immediate, preempting the ongoing transmission and avoiding the reception of invalid frames by the other stations. As a consequence all the receiving stations know that the frame being transmitted has an error. Thus, the transmitting station will automatically retry the transmission of the message. An Error Frame has the following structure:

e) 6-12 consecutive dominant bits (Error Flag). The station that firstly detects the error starts transmitting the Error Flag and hopefully every station will also recognise such error at the same instant. However, there is the possibility that other stations only recognise the bit stuffing error induced by the Error Flag. In this case, such stations will start transmitting Error Frames and thus the Error Flag will be 12 bits long;

f) 8 consecutive recessive bits (Error Delimiter) which signal the end of the error frame.

Concerning the available error detection and signalling mechanisms, the CAN protocol has the following capabilities:

a) Bit error: a transmitting station is continuously sensing the transmitted bits on the bus; if the observed bit does not corresponds to the transmitted bit, the transmitting station signals a transmission error (except if the bit error is observed during the identifier or the ACK Slot fields);

b) CRC error: the receiving station compares the CRC code of the received frame with its own evaluation of the CRC code. If the two CRC codes are different, the receiving station signals a transmission error. The CRC code can detect up to 5 randomly modified bits and up to 15 consecutively modified bits on the CAN frame.

c) Stuff bit error: Frames are transmitted with the insertion of stuff bits. That is, whenever there are more than five equal consecutive bits (up to the end of the CRC sequence), there is the insertion of an opposite bit in the frame. Whenever a receiving station detects more than five consecutive bits, it signals an error. Neither the Error Frame, nor the Overload Frame, are coded by the bit stuffing mechanism.

d) Form error: The receiving station verifies if the structure of the received frame is correct. If it is not, it signals a transmission error.

e) ACK error: The ACK Slot bit is used by receivers to acknowledge the correct (or incorrect) behaviour of the message transfer. Stations that have received a correct frame shall write a dominant bit on the ACK field. A recessive bit on this field may result from the absence of receiving stations or from a transmission error recognised from every receiver. If such error is verified, the transmitting station signals a transmission error.

f) Overload error: If the receiving station is not yet ready to receive another frame, it may transmit one or two consecutive Overload Frames (which have the same structure of the Error Frame) just after the end of the received frame.

g) Overload Frame form error: if the structure of the Overload Frame is not correct, an Error Frame is transmitted by the station(s) who detect such error.

Sending Error Frames is a very interesting mechanism to ensure that every station sees the same global state of the network (state coherence). However, it is possible that a failure in a station induces the transmission of consecutive error frames, blocking all the ongoing communications. To solve this problem, CAN controllers have two error counters (for transmitting and receiving errors, respectively) to isolate erratic stations. For instance, if a station is consecutively signalling errors in every Data/Remote Frame (e.g., due to a circuitry failure), there is a time bound after which the station can not signal any more error with active Error Flags. The values of these counters, which determine the operating state of the station, are increased or decreased (at different rates) as a function of the type of the detected error. These error counters acts as self-surveillance mechanisms, which disconnect faulty stations (fault-confinement techniques). There are three different operating modes:

a) Error-active, which is the normal operating mode.

b) Error-passive, where the station is still able to transfer / receive messages, but it must wait some time before initiating a transmission (automatically decreasing the transmission priority) and the error signalling is performed with passive Error Flags (6 consecutive recessive bits). When at this operation mode, the station can no longer interfere with frames transmitted by other stations.

c) Bus-Off, where the station is not able to transfer / receive messages.

3 Inaccessibility and Response Time Analysis in CAN Networks: Analysis of Previous Relevant Work

The use of CAN networks to support NCS applications requires not only time-bounded transmission services, but also a minimum level of confidence on the continuity of service. The integration of these two system requirements means that the temporary periods of network inaccessibility (periods during which stations are not able to use network services) must be considered for the response time analysis in CAN networks.

In this Section, we present some of the most relevant results concerning both the study of the inaccessibility characteristics and the analysis of message's response time in CAN networks.

3.1 Network and Message Models

We assume a network with *n* message streams defined as:

$$S_i = (C_i, T_i, D_i) \tag{1}$$

S_i defines a message stream *i* characterised by a unique identifier. A message stream is a temporal sequence of messages concerning, for instance, the remote reading of a specific process variable. C_i is the longest message duration of stream S_i. For the case of a Data Frame, this duration is the length of the frame itself. For the case of a Remote Frame, this duration includes the length of both the Remote Frame and the associated Data Frame. T_i is the periodicity of stream S_i requests. In order to have a timing analysis independent from the model of the tasks at the application process level, we assume that this periodicity is the minimum time interval between two consecutive arrivals of S_i requests to the outgoing queue. Finally, D_i is the relative deadline of a message; that is, the maximum admissible time interval between the instant when the message request is placed in the outgoing queue and the instant when either the message is completely transmitted (case of Data Frame) or the related response is completely received (case of Remote Frame).

3.2 Inaccessibility Analysis of CAN networks

Considering the available error detection and signalling mechanisms of the CAN protocol presented in Section 2.2, it follows that bit corruption errors can be detected by several of the CAN error detection mechanisms, such as CRC, stuff, form or ACK errors. From all these errors, the longest network inaccessibility [4] results from a Form Error detected at the end of the EOF delimiter. Such network inaccessibility is:

$$t_{ina\leftarrow form} = C_{MAX} + C_{error} + C_{IFS} \tag{2}$$

where C_{error} and C_{IFS} are the duration of an Error Frame and the Inter-Frame Spacing (two consecutive frames must be separated by at least 3 recessive bits), respectively, and C_{MAX} is the longest duration of a CAN message.

For the case of multiple consecutive bit errors, in [4] the authors consider the following failure assumption: *in a known and bounded time interval Trd, at most n transmissions can be affected by errors*. This assumption will also be used in Section 4, as the basis for the integration of inaccessibility in response time analysis of CAN networks.

Two different scenarios can be considered. Firstly, we can consider a burst of successive bit errors, where only the first one corresponds to a bit corruption in a Data Frame. The others will just disturb the Error Frame that is being transmitted in response to the first error. Another scenario, which results on an even longer duration for the network inaccessibility, considers that errors are sufficiently apart to interfere with *n* Data Frames, resulting in *n* failed attempts to transmit a Data Frame. The network inaccessibility resulting from this second scenario is:

$$t_{n\_ina} = n \times \left(C_{MAX} + C_{error} + C_{IFS} \right) \tag{3}$$

Apart from the frame error detection mechanisms, CAN controllers have two error counters to isolate erratic stations, preventing them from interfering with the bus operation (see Section 2.2). The values of these counters are increased or decreased (at different rates) as a function of the type of the detected error. In the case of a transmitter, the maximum number of transmission errors is given by:

$$n_{tx} = \left\lceil \frac{err_{thd}}{\Delta_{txerr}} \right\rceil \tag{4}$$

where err_{thd} is the error count threshold, and Δ_{txerr} is the increase of the counter at each detected transmission error. As the error count threshold is 127 and Δ_{txerr} is 8, then it can be 16 consecutive errors before a failed station enters into the error-passive state.

For the case of a receiver station, the maximum number of receiving errors is given by:

$$n_{rx} = \left\lceil \frac{err_{thd}}{\Delta_{rtxerr1} + \Delta_{rtxerr2}} \right\rceil \tag{5}$$

where Δ_{rxerr1} and Δ_{rxerr2} are used according to the type of the receiving error [4], to increase the receiving error counter at each detected error. For failed receiving stations, Δ_{rxerr1} is 8 and Δ_{rxerr2} is 1, then there can be at most 15 errors before the failed station enters into an error-passive state.

3.3 Response Time Analysis of CAN Networks

In [3] the authors addressed in detail the response time analysis of CAN networks. They assumed fixed priorities for message streams (since the network access is based on the identifier's priority and the message model assumes that each message stream has its own unique identifier) and a non-preemptive scheduling model (since lower priority messages being transmitted cannot be preempted by pending higher priority messages). Considering such scheduling model, they adapted existing schedulability analysis for task scheduling to the case of scheduling messages on a CAN network.

The worst-case response time of a queued message, measured from the arrival of the message request to the outgoing queue to the time the message is fully transmitted, is:

$$R_m = I_m + C_m \tag{6}$$

To guarantee that the system is schedulable it is sufficient to verify if every message has a response time smaller than its deadline. The term I_m represents the worst-case queuing delay - longest time interval between placing the message in the outgoing queue and the start of the message transmission.

The deadline monotonic (DM) priority assignment can be directly implemented in a CAN network, by setting the identifier field of each message stream according to the DM rule. Therefore, the worst-case queuing delay of message m is:

$$I_m = B_m + \sum_{\forall j \in hp(m)} \left(\left\lceil \frac{I_m + \tau_{bit}}{T_j} \right\rceil \times C_j \right) \tag{7}$$

where B_m is the worst-case blocking factor, which is equal to the longest duration of a lower priority message, and is given by:

$$B_m = \max_{\forall k \in lp(m)} \{0, C_k\} \tag{8}$$

The set $lp(m)$ is the set of message streams with lower-priority than message stream S_m. τ_{bit} is the duration of a bit transmission and $hp(m)$ is the set of message streams in the system with higher-priority than the message stream S_m.

4 Integration of Inaccessibility Issues with Response Time Analysis

In real-time applications, unexpected failures of the system are not acceptable, since value or timing requirements would not be met. It is clear that a real-time system must provide guarantees that deadlines are met, even in the presence of faults. Therefore, schedulability analysis must consider an expected set of error assumptions.

In this section, we integrate inaccessibility issues on the response time analysis of CAN networks. Essentially, we provide formulae to evaluate both the response time of messages and the resulting network load, considering a realistic assumption of a communication network disturbed by periods of inaccessibility.

4.1 Evaluation of a CAN Message Duration

A CAN message duration (Figure 2) can be evaluated considering that for each Data Frame there is a Data Field added to 44 bits of overhead (64 bits of overhead in CAN extended frames). Additionally, it must be considered the overhead concerning bit stuffing and Inter-Frame Spacing, and also the differences between Data Frames and Remote Frames (refer to Section 2).

Bit stuffing mechanisms are applied to the first 98 bits of the frame (it excludes the CRC delimiter, ACK and EOF fields), considering an 8 byte Data Field. In the worst case, bit stuffing increases the frame by $\lfloor 98/5 \rfloor = 19$ bits (23 bits in CAN extended frames), which means an overhead of 63 bits (87 bits in CAN extended frames), which is approximately 50% of the frame (58% in CAN extended frames).

A Remote Frame is similar to a Data Frame, without the Data field. Therefore, its maximum size is 44 bits (64 bits in CAN extended frames). As it is also coded by the method of bit stuffing, its size can be increased to 50 bits (74 bits in CAN extended frames). As the Remote Frame does not transfer data, we consider this frame as an overhead to the related Data Frame. Hence, a Data Frame that is a response to a Remote Frame has an overhead of 113 bits (161 bits in CAN extended frames), which is approximately 64% of the frame (72% in CAN extended frames).

Additionally, we need to consider the minimum Inter-Frame Spacing (IFS), which is 3 bits long, as a time interval during which the bus is not available for further transmissions. Also, if there is a slow controller on the bus, it may request extra time between frames, in order to process the received frame. In such cases, the controller is allowed to send two consecutive overload frames, preventing other stations from transmitting further frames. An Overload Frame has the same structure of an Error Frame (Section 2.2), and thus it means that with a slow controller on the bus, there is an extra overhead of 40 bits to be considered for every message.

4.2 Extending Tindell *et al.* [3] Analysis to consider Remote Frames

In [3], the use of Remote Frames is not considered. In our analysis, we consider a Remote Frame followed by the related Data Frame as a single transaction (which can be preempted between both frames), where these two frames have a precedence relation.

This means that the Remote Frame and the related Data Frame never try to simultaneously access the bus.

The response time analysis of the CAN network is equivalent to (6), (7) and (8), considering that:

a) The number of messages streams in the system is reduced, since a Remote Frame and its related Data Frame are considered as a single message stream;

b) The length of a message C_m includes the length of both the Remote Frame and the related Data Frame (Section 3.1);

c) The blocking that a Remote Frame and the related Data Frame produces in higher priority message streams can easily be identified, because these will be blocked either by the Remote Frame or by the related Data Frame, but not by both. As a Data Frame is always longer than the related Remote Frame, the blocking term B_m (8) is the maximum length of the Data Frames.

4.3 Response Time Analysis Considering Network Inaccessibility

In order to integrate the inaccessibility analysis presented in section 3.2 with the response time analysis of CAN message streams, two factors must be added to equations (6) and (7) to account for bus and station error. Ina_{bus} is the maximum inaccessibility time derived from bus errors and $Ina_{station}$ is the maximum inaccessibility derived from station errors.

$$R_m = I_m + C_m \tag{9}$$

$$I_m = B_m + \sum_{\forall j \in hp(m)} \left(\left\lceil \frac{I_m + t_{bit}}{T_j} \right\rceil \times C_j \right) + Ina_{bus} + Ina_{station} \tag{10}$$

To evaluate the inaccessibility time of a message *m*, it is necessary to derive how many errors can occur in the network while the message is waiting for transmission or is being transmitted. Notice that we follow the error assumption of [4], where *there can be at most n errors in an time interval T_{rd}*. Therefore, the number of errors that can interfere with the transmission of message *m* is given by:

$$n_{errors} = n \times \left\lceil \frac{t_m + C_m}{T_{rd}} \right\rceil \tag{11}$$

Hence, the inaccessibility time due to bus errors is:

$$Ina_{bus} = n_{bus\_errors} \times \left\lceil \frac{t_m + C_m}{T_{rd\_bus}} \right\rceil \times t_{ina \leftarrow form} \tag{12}$$

since the network inaccessibility due to a bus error is as defined in (2).

The maximum inaccessibility time due to a station error (transmitter or receiver errors, leading the station to the error-passive state) is a consequence of 16 consecutive transmission errors (Section 3.2). Therefore:

$$Ina_{station} = n_{station\_errors} \times \left\lceil \frac{t_m + C_m}{T_{rd\_station}} \right\rceil \times 16 \times \left(C_{MAX} + C_{error} + C_{IFS} \right) \quad (13)$$

4.4 Network Load Analysis Considering the Network Inaccessibility

The network load is given by the sum of the ratio transmission delay versus period of all message streams. Additionally, periods of network inaccessibility (due to on-going error detection and recovery mechanisms) must be also considered for the evaluation of the overall network load.

Considering that *in a time interval T_{rd} there are at most n errors*, each one inducing the maximum network inaccessibility, the network load resulting from such periods of network inaccessibility is:

$$U_{ina} = \frac{n_{errors} \times t_{ina \leftarrow form}}{T_{rd}} \quad (14)$$

As we must consider both bus errors and station errors, the overall network load is:

$$U = \left(\sum_{\forall m} \frac{C_m}{T_m} \right) + \frac{n_{bus\_errors} \times t_{ina \leftarrow form}}{T_{rd\_bus}} + \frac{n_{station\_errors} \times t_{ina \leftarrow form}}{T_{rd\_station}} \quad (15)$$

4.5 Case study (SAE Benchmark)

In this Section, we present the timing analysis of a CAN network example, where periods of network inaccessibility are considered. The chosen example is based on the SAE benchmark [5], which is used to evaluate different multiplexing communications technologies for the automotive industry. Although being specified for the automotive industry, the use of the SAE benchmark is an interesting option, since it allows the comparative analysis of the proposed methodology with previously available results [3].

This SAE benchmark specifies a set of messages that must be transferred between different subsystems in a prototype of an electric car, considering network date rates of: 125 Kbit/sec, 250 Kbit/sec, 500 Kbit/sec and 1 Mbit/sec. A simplification of this benchmark for the case of CAN networks was presented in [3], where the number of message streams is drastically reduced by piggybacking groups of data messages in single Data Frames, whenever that operation was possible. That simplification allowed for a reduction of the overall network load, due to the removal of the messages' overhead. Table 1 presents the resulting set of message streams, ordered by decreasing priorities.

Table 1 – SAE benchmark

Message	Size (bytes)	Period (ms)	Deadline (ms)
A	1	50	5
B	2	5	5
C	1	5	5
D	2	5	5
E	1	5	5

F	2	5	5
G	6	10	10
H	1	10	10
I	2	10	10
J	2	10	10
K	1	100	20
L	4	100	100
M	1	100	100
N	1	100	100
O	3	1000	1000
P	1	1000	1000
Q	1	1000	1000

In Table 2, we present the response time and the network load resulting from the message streams of Table 1 (evaluated using equations (9) and (15), respectively). In this table, we highlight all the message streams that may miss their deadlines. A network date rate of 125 Kbit/sec is considered (which leads to the highest network load) together with the following set of error assumptions:

a) from 0 to 4 bus errors in a 100 ms time interval, resulting from a bit error rate of approximately 10^{-4} (for a data rate of 125 Kbit/sec, this results in considering 0-4 errors within 12500 bits), which is an expectable bit error rate in aggressive environments;

b) a single station failure (burst of 16 consecutive bus errors), leading such station to the error-passive state. This assumption is consistent to the consideration of just a station failure during an extremely long period (some years).

Table 2 – Messages Response Times and Network Load (125 Kbit/sec)

Message	Response Time (ms)						Deadline (ms)
	0 errors	1 error	2 errors	3 errors	4 errors	Station error	
A	1,368	2,416	3,464	4,512	5,560	18,136	5
B	1,952	3,000	4,048	5,096	6,144	18,720	5
C	2,456	3,504	4,552	6,184	7,232	21,560	5
D	3,040	4,088	5,136	7,272	8,320	24,160	5
E	3,544	4,592	7,312	8,360	9,408	28,672	5
F	4,128	5,176	8,400	9,448	19,040	33,952	5
G	4,864	8,672	9,720	19,432	50,664	43,712	10
H	5,368	9,176	10,224	39,928		54,680	10
I	8,712	9,760	19,816			64,696	10
J	9,296	10,344	40,168			79,040	10
K	9,800	19,976				99,792	20
L	10,456	29,760				110,040	100
M	19,040	30,264				119,360	100
N	19,544	39,896				128,448	100
O	20,048	40,400				129,456	1000
P	28,632	50,032				129,960	1000
Q	28,656	50,056				129,984	1000
Network Load	81,19%	91,67%	102,15 %	112,63%	123,11%	81,192%	

As it can be seen, a set of message streams that is completely schedulable without considering periods of network inaccessibility (the 0 errors assumption of the 1st column is the assumption that was considered in [3]), is no longer schedulable even assuming low bit error rates. The simple consideration of one bit error per 100 ms time interval leads to a faulty timing behaviour in two of the message streams, and to an increase of more than 10% to the network load. If 2 bit errors are considered, there is a set of message streams (K-Q) which is no longer able to access the network, due to the exhaustion of the available bandwidth.

An interesting result of this analysis is that, conversely to what is common in priority driven systems, the first message stream to miss its deadline is not the lowest priority one, but one with an intermediate priority (message streams **F** and **J**). The reason for this abnormal behaviour is that the occurrence of a bus error results in the same inaccessibility period, whatever the message stream being considered. Therefore, message streams with smaller response-times will have the larger percentage increase on its message's duration, resulting that the most penalised message streams will be the ones with the smallest slack time (smallest difference between the message stream response time and its deadline).

This abnormal behaviour is present even in the case of errors during the transfer of lower priority messages. Thus, an *important conclusion* is that, in the presence of bus errors, a CAN fieldbus network is not able to provide different integrity levels, since errors in low priority messages interfere with the response time of higher priority messages. This result proves that the scheduling of messages in the presence of errors (which increase the network load) is not equivalent to the usual behaviour of fixed priority systems in overload conditions (where messages with lower priorities do not interfere with the response time of higher priority messages).

In Table 3, we analyse the same scenario for the case of a network data rate of 250 Kbit/sec. Obviously, as the duration of messages is reduced by 50%, the overall network load is also reduced by 50%. As a consequence, considering such reduced network load for this particular set of message streams (with harmonic periodicities), the message stream set is now schedulable for the considered error assumptions.

Table 3 – Messages Response Times and Network Load (250 Kbit/sec)

Message	Response Time (ms)						Deadline (ms)
	0 errors	1 error	2 errors	3 errors	4 errors	Station error	
A	0,684	1,208	1,732	2,256	2,780	9,068	5
B	0,976	1,500	2,024	2,548	3,072	9,360	5
C	1,228	1,752	2,276	2,800	3,324	9,904	5
D	1,520	2,044	2,568	3,092	3,616	10,992	5
E	1,772	2,296	2,820	3,344	3,868	11,828	5
F	2,064	2,588	3,112	3,636	4,160	12,624	5
G	2,432	2,956	3,480	4,004	4,528	13,576	10
H	2,684	3,208	3,732	4,256	4,780	14,272	10
I	2,976	3,500	4,024	4,548	5,072	14,816	10
J	3,268	3,792	4,316	4,840	6,744	16,780	10
K	3,520	4,044	4,568	5,092	6,996	17,324	20

L	3,848	4,372	4,896	6,800	7,324	17,652	100
M	4,100	4,624	6,528	7,052	7,576	17,904	100
N	4,352	4,876	6,780	7,304	7,828	18,156	100
O	4,604	5,128	7,032	7,556	8,080	18,408	1000
P	4,856	6,760	7,284	7,808	8,332	18,660	1000
Q	4,868	6,772	7,296	7,820	8,344	18,672	1000
Network load	40,596 %	45,836 %	51,076 %	56,316%	61,556%	40,596%	

Also included in Table 3 is the consideration of a single station failure. In this situation, higher priority messages miss their deadlines. It is interesting to notice that the response time of message stream **A** increases 13 times when a station error is considered, but the network load does not suffer any increase. That is due to the assumption of an extremely low failure rate for stations, leading to a negligible increase in the network load.

Finally, in Table 4 we analyse a different scenario, where there are no bus errors; instead, we consider that there is 1 station error for different network date rates. It can be seen that, even without bus errors, the message stream set is schedulable only at 1Mbit/sec, that is, it is only schedulable for a network load as low as 10%.

Table 4 – Message Response times and Network Load considering 1 station error

Message	Response Time (ms)				Deadline (ms)
	1 Mbit/sec	500 Kbit/sec	250 Kbit/sec	125 Kbit/sec	
A	2,267	4,534	9,068	18,136	5
B	2,340	4,680	9,360	18,720	5
C	2,403	4,806	9,904	21,560	5
D	2,476	4,952	10,992	24,160	5
E	2,539	5,496	11,828	28,672	5
F	2,612	5,768	12,624	33,952	5
G	2,704	6,098	13,576	43,712	10
H	2,767	6,224	14,272	54,680	10
I	2,840	6,370	14,816	64,696	10
J	2,913	6,516	16,780	79,040	10
K	2,976	6,642	17,324	99,792	20
L	3,058	6,806	17,652	110,040	100
M	3,121	6,932	17,904	119,360	100
N	3,184	7,058	18,156	128,448	100
O	3,247	7,184	18,408	129,456	1000
P	3,310	7,310	18,660	129,960	1000
Q	3,313	7,316	18,672	129,984	1000
Network load	10,149%	20,298%	40,596%	81,192%	

5 Pessimism Analysis

Up to this moment, a set of worst case error assumptions has been assumed. This results from guaranteeing timing requirements based on worst case conditions. It is, therefore, important to evaluate what is the pessimism inherent to the proposed approach. Considering the proposed analysis (equations (12) and (14)), some sources of inaccessibility-related pessimism can be identified:

a) It has been assumed that the worst case error assumptions are always present. That is, that all the $n_{errrors}$ are present in one round of messages;

b) It has been assumed that errors are always detected in the last bit of the longest Data Frame;

c) It has been also assumed that an Error Frame has always the maximum number of bits.

Although this set of worst case assumptions is necessary to the evaluation of the worst case response time of messages, it is also correct to say that they contain an important level of pessimism. In order to assess the impact of each one of these factors in the pessimism of the response time analysis, the following set of equations has been used:

$$Ina_{bus} = \mathbf{A}\times n_{bus\_errors}\times\left\lceil\frac{t_m + C_m}{T_{rd\_bus}}\right\rceil\times\left(\mathbf{B}\times C_{MAX} + (0.7+0.3\times\mathbf{C})\times C_{error} + C_{IFS}\right) \qquad (16)$$

$$U_{ina\_bus} = \frac{\mathbf{A}\times n_{errors}\times\left(\mathbf{B}\times C_{MAX} + (0.7+0.3\times\mathbf{C})\times C_{error} + C_{IFS}\right)}{T_{rd}} \qquad (17)$$

where **A** stands for the percentage of assumed errors in a period of T_{rd} (maximum of 4 errors), **B** stands for the percentage of the longest message to be transmitted and **C** is the percentage of the error frame length. As Error Frames have at least 14 bits, **C** can only be applied to the remaining 6 bits.

Figure 3 illustrates the impact of each of these factors on the network load and on the response time of Message **F** (Message **F** is chosen for the analysis, since it is the one with the smallest slack time, and one of the first two messages that misses its deadline). The variation of parameter **A** is made considering a value of 1 for parameters **B** and **C**. Variation of parameters **B** and **C** is made considering the existence of 3 bus errors.

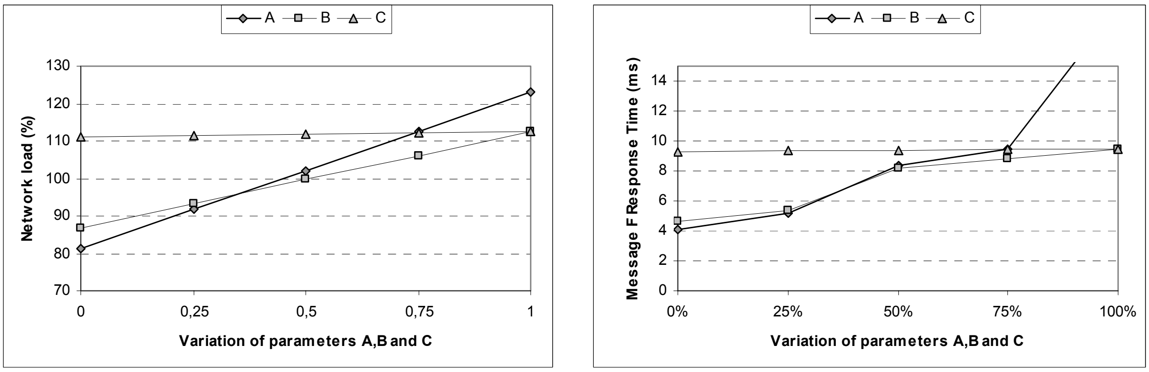

Figure 3 – Variation of the network load and message F response time with parameters A, B **and** C **(125 Kbit/sec)**

As it can be seen, the parameter that has the strongest influence in the set of messages is the considered bus error rate. The load of the network is largely penalised by an increase in the assumed error rate. An increase of 1 error increases the network load by approximately 13%.

The previous analysis showed that message stream **F** is only schedulable in the absence of errors (Table 2). In Figure 3, such non-schedulability of message stream **F** is reflected in the sudden increase of its response time, which is due to the increasing interference of message streams with 5 ms period. As shown in Figure 3, the response time of this message stream is highly dependent on the assumed error rate, and also on the assumed inaccessibility time caused by such errors. However, with smaller values for the inaccessibility time, the message stream is schedulable even for larger error rates.

In order to access the pessimism of considering that the error always occurs in the last bit of the largest message, Figure 4 shows the impact of parameter **B** for different bus errors assumptions.

Considering just one error, when parameter **B** is set to 0.5, the response time of message stream **F** will be just 4.744 ms, which compared to 5.176 ms (Table 2) gives a reduction for the response time of 8 %. This assumption is quite realistic since there is only one message that takes 6 bytes of data, and the majority of the messages have 1 or 2 bytes of data. Furthermore, for this scenario, message stream **F** becomes schedulable.

If greater error rates are assumed, the decrease of the response time is even more relevant. Network load can also decrease significantly if smaller inaccessibility times are assumed.

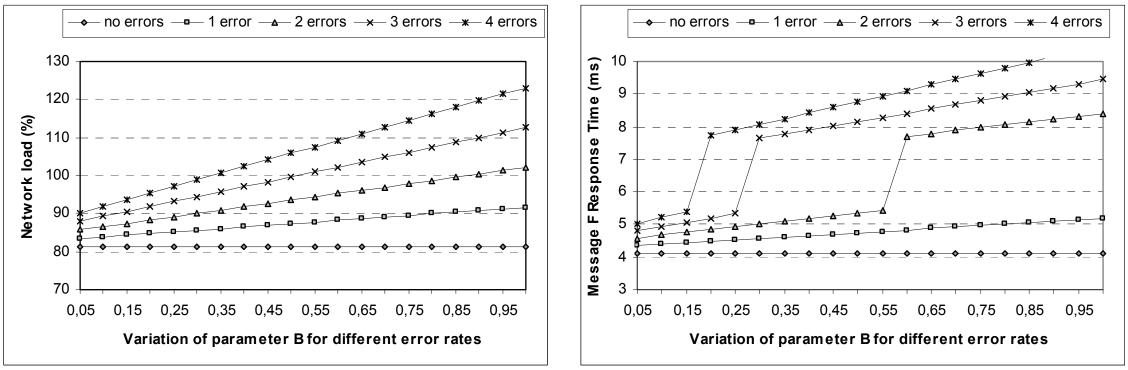

Figure 4 –Network load and message F response time, when varying parameter B for different error rates (125 Kbit/sec)

6 Conclusions

This paper addresses the integration of network inaccessibility issues with the response time analysis of CAN messages. It extends previous response time analysis, providing a more accurate analysis of the timing behaviour of CAN networks. A benchmark was used to illustrate the relevance of the proposed analysis and also to evaluate its inherent pessimism.

From the achieved results, it can be concluded that message streams with smaller response times will have the larger relative increase on its duration, due to network inaccessibility periods. Thus, the most penalised message streams will be the ones with the smallest slack time.

An important conclusion of the presented analysis is that, in the presence of bus errors, a CAN fieldbus network is not able to provide different integrity levels, since errors in low priority messages interfere with the response time of higher priority messages. Therefore, the scheduling of messages in the presence of errors (which increase the network load) is not equivalent to the usual behaviour of fixed priority systems in overload conditions (where messages with lower priorities do not interfere with the response time of higher priority messages). Another conclusion is that CAN is not resilient to station errors, since they can lead to large inaccessibility periods, thus making the system unschedulable.

The inherent pessimism of the proposed analysis has also been evaluated, and it is concluded that the network load and the message set response times' are highly dependent on the considered error rates and inaccessibility periods. It is also concluded that assuming lower inaccessibility periods, the system becomes schedulable even for greater bus error rates. This assumption is quite realistic, since the majority of the considered messages carry only 1 or 2 bytes of data.

7 References

[1] ISO 11898. (1993). Road Vehicle - Interchange of Digital Information - Controller Area Network (CAN) for High-Speed Communication. ISO.

[2] Zuberi, K. and Shin, K. (1997). Scheduling messages on Controller Area Network for Real-Time CIM Applications. In *IEEE Transactions on Robotics and Automation*, Vol. 13, No. 2, pp 310-314.

[3] Tindell, K., Burns, A. and Wellings, A. (1995). Calculating Controller Area Network (CAN) Message Response Time. In *Control Engineering Practice*, Vol. 3, No. 8, pp. 1163-1169.

[4] Rufino, J. (1995). A Study on the Inaccessibility Characteristics of the Controller Area Network. In Proc. of the 2nd International CAN Conference, London, United Kingdom, October 1995

[5] SAE. (1993). Class C Application Requirement Considerations. Technical Report J2056/1. SAE.