# Comparing Real-time Communication under Electromagnetic Interference

Ian Broster, Alan Burns
University of York, UK
{ianb, burns}@cs.york.ac.uk

Guillermo Rodríguez-Navas
Universitat de les Illes Balears, Spain
guillermo.rodriguez-navas@uib.es

## Abstract

*The contribution of this paper is threefold. First, an improvement to a previously published paper on the timing analysis of Controller Area Network (CAN) in the presence of transient network faults is presented. A probabilistic fault model is considered, where random faults from electromagnetic interference occur according to a Poisson distribution. The analysis provides worst case response times for message frames, not as a single value, but as a probability distribution. Secondly, a similar result is produced for time-triggered CAN (TTCAN), a version of CAN based on time-driven schedule. Thirdly, these analyses are applied to an example message set and used to discuss the dependability of event-triggered and time-triggered communication in the presence of electromagnetic interference. The results indicate that, an event-triggered bus can generally provide a higher probability of timely-delivery of data than a time-triggered bus.*

## 1. Introduction

The discussion about the relative merits of event-triggered and time-triggered communication has been long and varied [22, 11, 23]. A popular conclusion to the debate is that time-triggered communication (such as TTP [40, 21], TTCAN [15, 14]) is needed where dependable, real-time communication is required, while event-triggered communication (such as Controller Area Network, CAN [4, 18]) may be used elsewhere, where requirements for flexibility supersede any dependability requirements. This paper argues that event-triggered communication, and in particular CAN, is also suitable for dependable real-time communication. One reason for this is that event-triggered communication is able to provide a higher probability of timely delivery of message frames under electromagnetic interference than time-triggered communication alone can do. Supporting evidence appears later in this paper.

Proponents of time-triggered communication have argued that event-triggered communication is incapable of dependable real-time communication for the several reasons including [22, 14]: lack of fault containment (including the babbling idiot failure) and no guarantee of response times/deadlines.

Solutions to fault containment in event-triggered communication have been proposed, for example the Timely-CAN (TCAN[1]) protocol [7, 5] provides containment of delays on the bus, and there are at least three approaches to preventing babbling idiots [8, 39, 13].

In this paper, the supposed lack of guarantee of response times is considered in detail. We show that no electrical bus system (whether time-triggered or event-triggered) is capable of providing absolute guarantees of behaviour, however, CAN is able to provide a higher probability of successful timely delivery than, for example, Time-triggered CAN (TTCAN) [15].

The foundation of this argument is the axiom that faults are hard to predict; the precise times and nature of faults in the system are not known in advance. In particular, there is no bound on the number of individual faults that can occur within any time interval. This is discussed in more detail in Section 2.1. Since faults cannot be predicted, an event-triggered protocol that is able to respond to faults dynamically at run-time (by requesting retransmissions as necessary), may be able to provide a better service than a time-triggered protocol where the response to faults can only be considered in advance (for example in the case of TTP by routinely making two transmissions of each frame [21]).

This paper will first discuss the effect of faults in bus-based systems and some related work. A contribution of this paper is the probabilistic analysis in Section 3. Section 4 presents a similar analysis technique for TTCAN. These approaches are used in Section 5 to provide probabilities of failure an example message set on CAN and TTCAN. Section 6 briefly summarises the results.

## 2. Background

This section proceeds by discussing faults in communication, introducing CAN and TTCAN, and provides a model and a metric for comparison.

### 2.1. On Faults and Fault Models

Any electrical bus is susceptible to faults induced by electromagnetic interference (EMI). Screening and differential signalling are of benefit in reducing the ability for environmental radiation to affect the data on the bus, but no screening is perfect and in electrically noisy environments, the long lengths of cabling are always vulnerable

1 Originally termed LST-CAN.

[17]. However, the behaviour of a bus depends heavily on the individual environment, cable routing, cable type, proximity to potential sources of electrical noise and so on. Measuring and predicting the effects of EMI is difficult [24] because its causes are diverse (sparks, lightning, digital signals, radar, mobile phones [16], high voltage switching *etc.*) and uncertain (what is the worst/average effect of lightning?). There is no indication that the results of a given experiment are representative. Indeed, one would expect that the influence of sources such as mobile telephones, spark transmissions, radar and other devices are not constant from day to day, nor from system to system. The conclusion that this inevitably leads to is that faults cannot be predicted with any accuracy [17].

For the purposes of this paper, a *fault* is considered to have occurred when the state of the bus as read by one or more nodes is different to the state that was transmitted. These faults are typically caused by some form of electromagnetic interference from sources listed previously. Other types of fault such as network partitioning and faults in the nodes are not considered here, although it should be noted that many protocols have effective means of dealing with such faults [36, 32].

There have been several attempts to model interference in bus systems. Tindell [38] suggested that faults could be regarded as sporadic single-bit faults. That is, there is a minimum separation between faults. Therefore in CAN, faults can be treated in almost the same way as frames in the sporadic stream model. Additionally, to allow for the possibility of faults occurring closer than the minimum separation, a single burst of faults is accepted (and the burst has a maximum length).

The model is simple and effective but somewhat crude. There is no evidence that it reflects the nature of real faults, although some simulation work [5] suggests that its use is not wholly inappropriate. In the absence of the ability to accurately measure or predict faults, this model is useful in that it gives a *guide* as to how much spare bandwidth is required to meet deadlines—a “fudge factor”. However, in use, it quickly leads to a very high overhead in the analysis which is not necessarily justified in practice.

One fault model frequently used to describe EMI in other domains is to model faults as a random pulse train with an exponential distribution of inter-arrival times, forming a Poisson distribution [10, 19]. This means that there is neither a guarantee on the minimum separation between faults, nor on the number of faults within an interval. This model is well suited to modelling faults in electronic circuits and networks.

Navet [27] adopted a probabilistic fault model for CAN using a Poisson distribution. Faults are modelled as a stochastic process which considers both the frequency of the faults and their gravity. In that model, faults in the channel occur following a Poisson law and can be either single-bit faults (which have a duration of one bit) or burst errors (which have a duration of more than one bit) according to a random distribution.

Note that if the occurrence of faults in the channel follows a Poisson distribution, the maximum number of transmission errors suffered by the system in a given interval is not bounded, so the probability of having sufficient interference to prevent a message from meeting its deadline is always non-zero; therefore every system is inherently unschedulable (as should be expected).

A probabilistic model encapsulates the idea that there may normally be a low level of faults—well separated with large inter-arrival times—yet occasionally, the system may experience higher loads with faults occurring closer together. Of course, a random distribution, particularly a Poisson distribution (which requires an assumption of independent faults) cannot be a precise measure of fault activity. Nevertheless, we hypothesise that in the absence of the ability to perform precise measurement for general system, a Poisson distribution is an appropriate way of modelling faults, with significant merit. Its use is proposed in replacement of the sporadic fault model which is the most frequently used scheme in industrial practice.

However, previous approaches to providing probabilities of deadline failure [12, 27] are shown to give quite pessimistic results [9, 5].

## 2.2. Event- and Time-triggered Communication

This paper considers two general forms of communication, event-triggered and time-triggered. This section briefly describes each one, and its behaviour in the presence of faults.

Time-triggered protocols, such as TTP and TTCAN, make scheduling decisions based on the progression of time. The usual method is to use a TDMA (Time Division Multiple Access) scheme, where bus access is determined entirely by predefined time-slots. Generally, TDMA schemes divide time into repeating *cycles* (sometimes also into major cycles and smaller minor cycles), each node is allowed to write to the bus only at certain times within a cycle. In order to ensure that collisions cannot occur, each node is given a different time *slot* in which to transmit. The slots and cycles are determined off-line, often using tool support. The cycles are usually the same length and repeat indefinitely, making the bus *periodic* in nature. For detailed descriptions of TDMA approaches, there are numerous descriptions [21].

Event-triggered real-time protocols, of which CAN [4] is a notable example, make scheduling decisions based on which nodes wish to transmit at any given time. CAN achieves *non-destructive collision resolution*: when a collision occurs between two or more frames, the frame with the highest priority continues transmitting and any other nodes stop transmitting their lower priority frames (the means by which this is achieved is explained elsewhere [4, 18, 5]). Ultimately, CAN provides a non-preemptive, priority-based bus scheduling protocol. One key feature of CAN is that if a fault occurs during transmission of a frame (hence corrupting the frame) then the frame is automatically queued for retransmission. This can generally be used to provide an *as-*

*sured delivery* service [26, 30, 33], but can also be the cause of uncertainty in *timing* since the number and distribution of faults determines the time at which a frame is received.

Many buses based on a time-triggered paradigm (such as ARINC-629 [1], TTP, FlexRay [3]) do not make any response to corrupted frames, other than to disregard the frames. These protocols leave all the concerns of absent data to the application. The result of this is that the *timeliness* of the bus is preserved, but the number of messages that are *lost* is related to the number and distribution of faults that occur.

TTCAN is a recent proposal to extend CAN by adding a time-triggered higher-level protocol. Thus, in principal at least, one may achieve time-triggered communication using existing, readily available and inexpensive CAN hardware. It is now accepted as part 4 of ISO standard 11898.

### 2.3. Deadlines and Timing Analysis

It is common to talk of hard deadlines in communication. However, in this paper we accept the unpredictable nature of faults [17], and as Section 2.1 explained, no matter what form of protocol is used, if faults may occur without constraint then absolute guarantees of timely delivery cannot be made. Instead, it is necessary to accept that some frames may either arrive late (in the case of CAN) or not at all (in the case of TTP, TTCAN and TCAN). Therefore we adopt the firm deadline model, which carries the implication that occasional delivery failures are acceptable and expected. This practical approach is necessary; guarantees of hard deadlines *cannot* be made in any form of electrical communication with the potential for unpredictable faults.

Timing requirements in dependable real-time communication are well described by firm deadlines. Firm deadlines are easy to reason about because they allow the creation of a precise boundary point between correct and incorrect behaviour, at which point error recovery actions may be necessary. Further, timeliness may be tested easily during the lifetime of the system: a deadline was either met, or it wasn't; this allows testing procedures to be formalised and the justification of confidence in a system's correctness.

We may define a successful message delivery as when a message is received by an appropriate set of nodes (perhaps all nodes) before its (specified) deadline. In a fairly static system, as many dependable systems are, is reasonable to assume that a receiving node has sufficient knowledge of the time to be able to determine whether or not a message is received in a timely fashion. Therefore no distinction is made between a message that is permanently lost and one which is delivered after its deadline; they are of the same 'value' to the application.

In this domain, periodic streams of traffic are common practice. Additionally, time-triggered protocols are most suited to periodic traffic. Therefore, it is appropriate to assume the periodic stream model of traffic for the comparison in this paper.

Timing analysis of time-triggered protocols is straightforward. Since the time of each message is determined in advance, it is simple to calculate response times and hence determine whether deadlines will be met. This can be done off-line, to determine guarantees of behaviour.

Timing analysis of event-triggered protocols is more complicated, although no less understood. Worst case response times for CAN are calculated in a similar way to a fixed priority scheduler [38] to provide deadline guarantees.

It should be noted that in the absence of any kind of fault (whether EMI related or not), the deadline guarantees provided by both types of bus protocol are of identical significance. If no faults occur then both bus architectures can equally provide guarantees that all messages are received (assured delivery) and received on time (timely delivery).

However, it is clear that in a dependable system, it is not appropriate to assume that there are no faults. Indeed, Section 2.1 argues that at the very least, EMI-related faults cannot be overlooked. Instead, this paper promotes the concept of probability of successful delivery. This allows a convenient method to compare the reliability (with respect to timely delivery) of protocols with differing failure semantics.

### 2.4. Related Work on CAN

The first timing analysis of CAN was presented by Tindell [38]. His model consists of periodic messages and a sporadic faults. The work forms the foundation of significant later work and this paper. Tindell's equations are presented below, because they will be used later in the probabilistic analysis. They are changed slightly from their original form to remove one slight discrepancy and to aid mathematics later; for details of the changes, see [5]. The worst case response time of a data frame $i$ is given by $R_i$ in equations (1) and (2).

$$R_i = J_i + t_i \tag{1}$$

$$t_i = B_i + C_i + I_i(t_i) + E_i(t_i) \tag{2}$$

where: $C_i$ is the worst case transmission time [31] (the time it takes, in the worst case to send frame $i$) assuming no errors, maximum bit stuffing and not including the inter-frame space that follows the frame,

$$C_i = \left(44 + 8b + \left\lfloor \frac{34 + 8b - 1}{4} \right\rfloor \right) \tau \tag{3}$$

where $\tau$ is one bit-time and $b$ is the number of data bytes in the frame (0 to 8). Due to bit-stuffing, the actual length of a frame depends on the data and arbitration value. For an 8-byte data frame, for example, the frame size can vary between 108 and 126 bits. $C_i$ is understood to be the maximum possible length of a frame since this is a worst case analysis. A study on the actual amount of bit stuffing that is typically observed provides a useful method to reduce its effects [28].

The worst case blocking, $B_i$ is the maximum time a message may need to wait due to a lower priority message on the bus:

$$B_i = \max_{\forall k \in \mathsf{lp}(i)} (C_k) + S \tag{4}$$

where $\mathsf{lp}(i)$ is the set of messages with lower priority than $i$. Note that if $i$ is the lowest priority frame then $B_i = S$.

The term $I_i(t)$ is the worst case interference that message $i$ may receive in $t$ time units:

$$I_i(t) = \sum_{j \in \mathsf{hp}(i)} \left\lceil \frac{t - C_i + J_j + \tau}{T_j} \right\rceil (C_j + S) \tag{5}$$

where $\mathsf{hp}(i)$ is the set of messages with higher priority than $i$ and $J_j$ is the worst case release jitter of frame $j$. Note that the numerator in the fraction involves $t - C_i$ as interference can only take place before frame $i$ begins transmission (care must be taken never to require $I_i(a)$ if $a < C + J + \tau$). The $\tau$ is used to eliminate 'edge effects' in non-preemptive analysis where a high priority frame becomes ready as a medium priority one completes [2].

Bus faults are incorporated into the analysis by considering their effects. $E_i(t)$ is the worst case overhead due to network faults and extra frames that can occur in any given time interval, $t$. $E_i(t)$ must be bounded and non-decreasing over $t$. Each fault on CAN will cause an error frame to be transmitted and the retransmission of either a higher priority frame or if the fault falls in frame $i$ then that frame will have to be retransmitted before it is received. In the worst case (*i.e.* where the fault falls on the last bit of a frame) then a fault leads to an overhead of the error frame and the retransmission (in addition to the the whole of the lost frame, which is already considered in the WCRT formulation). The sporadic fault model derives the following equation for $E_i(t)$:

$$E_i(t) = \left( n_{burst} + \left\lceil \frac{t}{T_F} \right\rceil \right) \left( \max_{j \in \mathsf{hep}(i)} C_j + E \right) \tag{6}$$

where $T_F$ is the minimum inter-arrival-time between faults and $n_{burst}$ is the maximum number of faults that can occur in succession during a burst, $E$ is the maximum length of an error frame (taken to be $29\tau$), $\mathsf{hep}(i)$ is the set of streams of higher or equal priority than $i$. The leftmost product-term of equation (6) is the maximum number of faults that can occur in an interval $t$, and the rightmost product-term is the maximum overhead of each fault.

Equation (2) may be solved iteratively by forming a recurrence relation with $t_i^0 = C_i$ which terminates when $t_i^{n+1} = t_i^n$ or fails when $t_i^{n+1} > D_i - J_i$ where $D_i$ is the deadline and $D_i \leq T_i$. If there is a solution $R_i \; \forall i$ and $R_i \leq D_i$ then the analysis above will guarantee that all messages will always meet their deadlines, provided that there are no faults.

## 2.5. Tree-based Probabilistic Analysis

A previous publication [9] presented an approach to producing a probability distribution of response times based on a random fault model. This approach was very general and may be applied in a number of situations. Its use was shown to be beneficial and accurate. However, it was quite computationally expensive and does not necessarily cover all the search space. The general approach was to produce a probability tree of feasible scenarios starting from a critical instant, then to traverse the tree to calculate a distribution of response times. Branch pruning, based on a threshold parameter for insignificant probabilities, limited the size of the tree. The approach therefore requires a compromise between the time taken to compute the analysis and the accuracy of the analysis results.

The basis of the algorithm are equations (1) and (2). For the error overhead, $E_i(t)$, given in equation (8), a Poisson probability distribution is used, derived as follows.

The general form of the Poisson distribution, considering $m$ events in time $t$, is:

$$\mathrm{p}_t(F = m) = \frac{e^{-\lambda t}(\lambda t)^m}{m!} \tag{7}$$

It is assumed that each fault causes the maximum length error frame ($E$) and occurs on the last bit of the longest frame, such that the maximum overhead due to one fault is equal to:

$$M_i = E + \max_{\forall j \in \mathsf{hep}(i)} C_j$$

where $\mathsf{hep}(i)$ is the set of messages with higher or equal priority to $i$. Therefore the error overhead function, $E_i(t)$, is a random distribution:

$$E_i(t) = mM_i \text{ with probability } \mathrm{p}_t(F = m) \tag{8}$$

The immediate result of the analysis is a set of pairs $R_i = \langle t_i, \mathrm{p}(t_i) \rangle$. More usefully, a cumulative probability distribution can be plotted.

# 3. New Analysis of CAN

In the specific case of faults on CAN, the probabilistic analysis above [9] can be modified to reduce the computation time of the algorithm and eliminate branch pruning to achieve complete coverage. Section 3.1 first makes a small improvement by pre-computing response times, then Section 3.3 exploits the specific fault model to produce a scheme which has a very low computation cost and achieves full coverage.

## 3.1. Pre-computing Response Times

As previous examples [9] and one later in this paper illustrate, the shape of the probability distribution output is 'stepped'. The cause is the simple nature of the error overhead function, equation (8). It is noted, therefore, that there are only a relatively small number of possible worst case response times. A large number of different scenarios contribute to the probability of each response time value; the probability of each response time is the sum of the probabilities of these scenarios.

Therefore, it is possible to pre-compute the set of possible response times up to some point (such as the period of the message, which is the limit of the analysis) and then calculate the possible scenarios which contribute to each response time. Note that we use the notation $R_{i|K}$ to mean the worst case response time given that $K$ faults delay the frame.

Pre-computing the response times is done in the expected manner, by forming a recurrence relation from equation (9).

$$R_{i|K} = B_i + C_i + I(R_{i|K}) + E_{i|K}(R_{i|K}) \tag{9}$$

where

$$E_{i|K}(t) = K(E + \max_{\forall j \in \mathsf{hep}(i)} C_j) \tag{10}$$

### 3.2. Scenarios

After pre-computing the possible worst case response times, it is necessary to consider the scenarios that contribute to each possible value. In this paper, an analysis scheme is presented which avoids having to enumerate all possible scenarios, hence avoid a large, expensive tree traversal. The following discussion of these scenarios is useful to aid understanding of the scheme.

Equation (1) generates a set of non-overlapping intervals, as shown in Figure 1. The notation $e(K)$ is used to denote the number of faults that occur in time interval $(R_{i|K-1}, R_{i|K}]$ (or $(0, R_{i|K}]$ where $K = 0$).

| 0 | $R_{i\|0}$ | $R_{i\|1}$ | $R_{i\|2}$ | $R_{i\|n-1}$ | $R_{i\|n}$ |
|---|---|---|---|---|---|
| $e(0)$ | $e(1)$ | $\cdots$ | $e(n-1)$ | | |

**Figure 1. Possible Worst Case Response Times for a Given Number of Faults.**

Using the shorthand, [210] to mean the scenario e(0)=2, e(1)=1, e(2)=0, Table 1 shows the scenarios which contribute to a given response time. Note that (for example) the sequence [1020] cannot contribute to $R_3$, since the sequence begins [10] which contributes only to $R_{i|1}$ because at time $R_{i|1}$, there has been only one fault therefore the iteration of the WCRT equation terminates. It would be pessimistic to attach the scenario [1020] to the probability of the response time for 3 faults, $R_{i|3}$ as a different approach does [27].

| Response Time | Possible Scenarios (Shorthand) | Number of Scenarios |
|---|---|---|
| $R_{i\|0}$ | [0] | 1 |
| $R_{i\|1}$ | [10] | 1 |
| $R_{i\|2}$ | [200], [110] | 2 |
| $R_{i\|3}$ | [3000], [2100], [2010], [1200], [1110] | 5 |
| $R_{i\|4}$ | [40000], [31000], [30100], [30010], [21100], [21010], ... | 14 |

**Table 1. Enumeration of Scenarios.**

The sequence of the number of scenarios which constitute each response time grows rapidly. It begins 1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, 58786, 208012, 742900, and is known as the Catalan Series [35] given by the formula:

$$\frac{(2K)!}{K!(K+1)!} \tag{11}$$

where $K$ is the number of faults, as in $R_{i|K}$. This growth gives some understanding of the potential size of the tree in the previous approach, although the tree never actually reaches anywhere near it potential maximum size because of the branch pruning.

### 3.3. Efficient Probabilistic Analysis

From the precomputed response times, an efficient probabilistic analysis can be used to find the probabilities of the response times without enumerating all the scenarios. The technique is presented in this section.

**3.3.1. Notation** For clarity, the notation $\mathrm{p}(n,t)$ is used to denote the probability of $n$ faults occurring in interval $t$. $\mathrm{p}(n,t) \equiv \mathrm{p}_t(n)$, as defined in equation (7). The notation $\mathrm{p}(R_{i|K})$ is the upper bound on the probability that a frame $i$ is affected by exactly $K$ faults and hence may arrive no later than $R_{i|K}$. $R_{i|K}$ is precomputed as shown in Section 3.1.

The analysis will be derived by considering the scenarios which contribute to each particular response time.

**3.3.2. Calculating $\mathrm{p}(R_{i|0})$** Considering the worst case response time with no faults, $R_{i|0}$: $\mathrm{p}(R_{i|0})$ is the upper bound on the probability of faults not causing a frame $i$ to exceed this time. It is simply the probability that there are no faults in the interval $(0, R_{i|0}]$. As Table 1 showed, this is the only possible scenario that can produce a response time of $R_{i|0}$. This is exactly the same as the previous analysis.

$$\mathrm{p}(R_{i|0}) = \mathrm{p}(0, R_{i|0})$$

**3.3.3. Calculating $\mathrm{p}(R_{i|1})$** For the response time $R_{i|1}$, *i.e.* 1 fault, there is only one scenario which can cause this. Table 1 shows this to be [10], there must be exactly one fault in $(0, R_{i|0}]$ and no faults in $(R_{i|0}, R_{i|1}]$. The previous section suggested that the probability of [10] may be calculated by summing the probabilities of the scenarios (just one in this case).

However, an alternative approach is to begin with the probability of having exactly one fault in the interval $(0, R_{i|1}]$, which is $\mathrm{p}(1, R_{i|1})$. This can occur in only two ways: [01] or [10], of which only [10] is of interest. The probability of scenario [01] is already partially calculated because this is $\mathrm{p}(R_{i|0})$ multiplied by the probability of 1 fault in $(R_{i|0}, R_{i|1}]$.

$$\begin{aligned} \mathrm{p}(1, R_{i|1}) = {} & \mathrm{p}(R_{i|1}) && [10] \\ & + \mathrm{p}(R_{i|0})\,\mathrm{p}(1, R_{i|1} - R_{i|0}) && [01] \end{aligned}$$

Hence:

$$\mathrm{p}(R_{i|1}) = \mathrm{p}(1, R_{i|1}) - \mathrm{p}(R_{i|0})\,\mathrm{p}(1, R_{i|1} - R_{i|0})$$

**3.3.4. Calculating** $\mathrm{p}(R_{i|n})$ Likewise, to calculate $\mathrm{p}(R_{i|2})$, is it possible to begin with the probability that there must be exactly two faults in $(2, R_{i|2}]$ and then exclude the scenarios where there were exactly 0 faults in $(0, R_{i|0}]$, or exactly 1 fault in $(0, R_{i|1}]$ since these scenarios would give rise to smaller response times.

$$\begin{aligned}\mathrm{p}(R_{i|2}) = &\mathrm{p}(2, R_{i|2}) \\ &- \mathrm{p}(R_{i|1})\,\mathrm{p}(1, R_{i|2} - R_{i|1}) \\ &- \mathrm{p}(R_{i,|0})\,\mathrm{p}(2, R_{i|2} - R_{i|0})\end{aligned}$$

The result is generalised as follows. The probability of exactly $n$ faults in $R_{i|n}$ is derived directly from the Poisson distribution equation, $\mathrm{p}(n, R_{i|n})$. However only some permutations of faults can possibly lead to such a response time. The permutations which cannot lead to $R_{i|n}$ are those which would lead to a response time $R_{i|j}$ where $j < n$.

If there are $j$ faults in $(0, R_{i|j}]$ then (because there are $n$ faults in $(0, R_{i|n}]$) there must be $n - j$ faults in $(R_{i|j}, R_{i|n}]$. So, the probability of $j$ faults in $(0, R_{i|j}]$ given that there are $n$ faults in $(0, R_{i|n}]$ is $\mathrm{p}(R_{i|j})\,\mathrm{p}(n - j, R_{i|n} - R_{i|j})$. This value can then be subtracted from the probability $\mathrm{p}(R_{i|n})$.

The resulting general equation for the upper bound on the probability of worst case response time $R_{i|n}$ is:

$$\begin{aligned}\mathrm{p}(R_{i|n}) = &\mathrm{p}(n, R_{i|n}) \\ &- \sum_{j=0}^{n-1} \mathrm{p}(R_{i|j})\,\mathrm{p}(n - j, R_{i|n} - R_{i|j})\end{aligned} \tag{12}$$

Results obtained from this scheme should also have greater accuracy than the tree based probabilistic schemes because there are far fewer calculations, hence less accumulated rounding errors. Additionally, it should be noted that there is no branch pruning; therefore full coverage should be achieved.

Implementation of this is trivial, so code is not shown. A software implementation based on equations (9) and (12) was used to perform the study in Section 5.

The results of the analysis are of the same form as the previous analysis, and are usefully plotted as a cumulative distribution of response times. The only difference with the previous analysis is that when using an analysis based on pre-computed worst case response times (rather than the general tree-based form of the analysis) is that it leads to a difficulty in determining the accuracy. Response times greater than the deadline are indistinguishable from loss of precision from *e.g.* floating point arithmetic. Therefore, it is not possible to check the accuracy directly. However, in general, since uncovered regions are added to the probability of being unschedulable, this does not affect the overall results of the analysis.

Finally, the probability of deadline failure for frame $i$ is given by equation (13).

$$\mathrm{p}_i(failure) = 1 - \sum_{\forall K | R_{i|K} < D_i} \mathrm{p}(R_{i|K}) \tag{13}$$

# 4. Analysis of TTCAN

In this section, a similar probabilistic analysis for TTCAN is presented. The same Poisson fault model will be assumed. The analysis is somewhat simpler than the CAN analysis, since the times of the messages are predetermined.

For any given message, $i$, of maximum length $C_i$, the worst case probability of it being lost is the probability of it being hit by one or more faults, which is 1 minus the probability of it not being hit by a fault. The Poisson equation (7), directly derives the probability of an unsuccessful delivery.

$$\begin{aligned}\mathrm{p}(k \neq 0) &= 1 - \frac{e^{-\lambda C_i}(\lambda C_i)^0}{0!} \\ &= 1 - e^{-\lambda C_i}\end{aligned} \tag{14}$$

Where a TTCAN schedule contains more than one instance of a frame for fault-tolerance, then since we assume faults are independent, the probability of an unsuccessful delivery in $n$ attempts is 1 minus the probability of $n$ failures:

$$\mathrm{p}_i^n(failure) = (1 - e^{-\lambda C_i})^n \tag{15}$$

# 5. Comparison

There are several well-known message sets used for evaluation of CAN. The Society of Automotive Engineers (SAE) produced a set of messages [34] that might be transmitted within a vehicle. This message set has frequently been used as a benchmark to evaluate CAN and has been the source of some discussion [37, 20]. Results using this benchmark and a probabilistic analysis have been previously published [9, 5], and have shown positive evaluation of the resilience of event-triggered communication with faults. To avoid repetition of previous results, a new dataset is presented and used to evaluate the probabilistic analyses in this paper and hence compare the behaviour of event-triggered communication and time-triggered communication in the presence of faults.

The message set is from a design for a simple mobile robot using "smart sensors" (sensors which contain an integral CAN controller). There are only 6 messages in this example, but they are quite demanding, both in terms of frequency of transmission and short deadlines. The bus is used at 256 kbit/s baud-rate, making the total utilisation of this set fairly low, 41%. Such a low value is chosen in order to be able to construct a TTCAN schedule where all messages can be transmitted twice. The messages appear in Table 2 with conventional worst case response times from equation (1). Note that we adopt the convention that a higher priority is indicated by a higher numeric priority, rather than stating CAN identifiers (where a lower numeric value indicates a higher priority). For the fault model, a value of $\lambda = 30\,\mathrm{faults/second}$ is chosen. This value of $\lambda$ is used because it has been frequently used in the past [27, 29] as an expected number of faults in an aggressive environment.

In order to be able to construct a comparison between CAN and a TDMA bus, the periods of these messages have

| Priority | Length ($\mu$s) | Period ($\mu$s) | Deadline ($\mu$s) | WCRT | Comment |
|---|---|---|---|---|---|
| 6 | 288 | 2000 | 2000 | 828 | MotorCtrl |
| 5 | 328 | 4000 | 4000 | 1168 | Wheel1 |
| 4 | 328 | 4000 | 4000 | 1508 | Wheel2 |
| 3 | 528 | 8000 | 8000 | 2048 | RadioIn |
| 2 | 248 | 12000 | 12000 | 2608 | Proximity |
| 1 | 528 | 240000 | 240000 | 2320 | Logging |

**Table 2. Example Message Set**

been selected to be harmonic. This is of course not necessary for an event-triggered bus and can even limit the performance compared to when non-harmonic periods are used [6].

The immediate result of the CAN probabilistic analysis with $\lambda = 30\,\text{faults}/\text{second}$ appears in Figure 2 as a cumulative probability distribution. The graph shows the probabilities of each message exceeding a given response time. The lowest point on each line can be interpreted as the probability of deadline failure. The probabilities of deadline failures are recorded for later comparison.

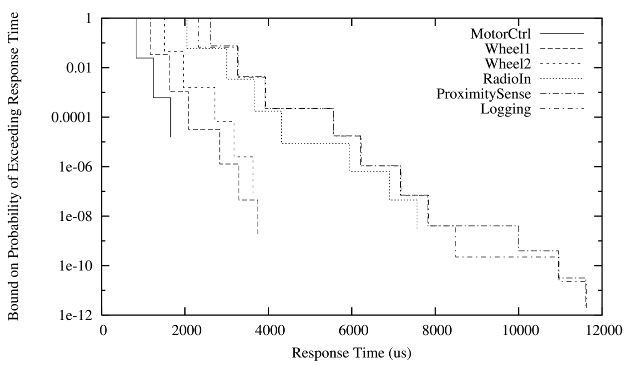


**Figure 2. Probabilistic Analysis of Message Set on CAN at $\lambda = 30$**

An equivalent schedule for TTCAN was designed, using a cycle period of 240 ms. Concerns about jitter, offsets and synchronisation with nodes were not considered in the design of the cycle. Since the utilisation is less than 50% there is sufficient space to transmit all frames twice. Two TTCAN experiments are considered, one where each frame appears in the schedule once, and the other where each frame appears in the schedule twice. The probabilities of deadline failure for the message sets appear in Table 3.

From the table, it is clearly seen that in all cases, CAN outperforms TTCAN, in most by several orders of magnitude. This is an intuitive result. CAN uses its flexibility to retransmit frames only when necessary, making efficient use of the bandwidth. There is no limit on the number of times that any given frame is retransmitted; if it is corrupted on the third attempt then a fourth is made, and so on. Figure 2 gives an indication about how many message retransmissions are used to provide this level of probability; each

| Priority | Probability of Failure | | |
|---|---|---|---|
| | CAN | TTCAN[1] | TTCAN[2] |
| 6 | $1.5 \cdot 10^{-5}$ | $8.6 \cdot 10^{-3}$ | $7.4 \cdot 10^{-5}$ |
| 5 | $1.6 \cdot 10^{-9}$ | $9.8 \cdot 10^{-3}$ | $9.6 \cdot 10^{-5}$ |
| 4 | $8.7 \cdot 10^{-8}$ | $9.8 \cdot 10^{-3}$ | $9.6 \cdot 10^{-5}$ |
| 3 | $2.7 \cdot 10^{-9}$ | $1.6 \cdot 10^{-2}$ | $2.5 \cdot 10^{-4}$ |
| 2 | $2.1 \cdot 10^{-12}$ | $7.4 \cdot 10^{-3}$ | $5.5 \cdot 10^{-5}$ |
| 1 | $< 1 \cdot 10^{-20}$ | $1.6 \cdot 10^{-2}$ | $2.5 \cdot 10^{-4}$ |

**Key**

[1] Scheduled once
[2] Scheduled twice

**Table 3. CAN and TTCAN, Prob. of Failure**

'step' in the graph is caused by one more fault and hence one more retransmission (the retransmission does not necessarily occur in the frame that is under investigation). Considering the frame with priority 5, to achieve a probability of failure of $1.6 \cdot 10^{-9}$ it tolerates 5 faults, occurring while it is queued or being transmitted.

The effect of retransmission is to delay all lower priority frames, but as the example demonstrates the probability the fault propagating downwards to the extent that these frames are also pushed beyond their deadlines is low. On the other hand, TTCAN is constrained to 2 transmissions, regardless of whether more are needed or not, hence creating a bound on the probability of delivery.

Other datasets (including the SAE benchmark) not shown in this paper demonstrate similar results. Accurate simulation and fault injection [5] provide further evidence that the probabilistic analysis is both accurate and exhibits low pessimism.

Despite these results however, it is not necessarily the case that the analysed probabilities of failure for CAN are always lower than TTCAN. It is possible to contrive a message set with very short deadlines whereby some medium priority frames have a worst case probability of failure with CAN to be higher than an equivalent TTCAN[2] (scheduled twice) analysis result. The characteristic of such a message set is that the CAN WCRT of one transmission, one retransmission and the worst case blocking is beyond the deadline: $C_t + S + C_i + E + B_i > D_i$ so only one transmission is useful. Whereas in a TTCAN schedule in a system with synchronised (time-triggered) nodes, where one would not normally incorporate the blocking in timing analysis (because nodes are tightly synchronised) can accommodate two transmissions, $C_t + S + C_i + E < D_i$.

## 6. Conclusion

Based on the assumption that faults are difficult to predict, this paper has promoted a Poisson fault model. It is accepted that this model cannot be expected to be exact, but it is proposed as a reasonable model for design guidance and to gain an understanding of the behaviour of the system. Using this fault model, schemes for simple probabilistic analysis for CAN and TTCAN are explained.

For both protocols, the analyses provide probabilities of successful delivery of frames before a firm deadline. Additionally, for CAN, the analysis gives a probability distribution of worst case response times which derives the probability for deadline failure.

This result applied to a simple message set, which is used as evidence to compare CAN with TTCAN. Using identical fault models, and the metric 'probability of timely delivery', the results showed that CAN outperformed TTCAN in terms of probability of successful delivery.

There are a broad range of motivations when choosing a protocol, of which the ability to tolerate transient faults is only one. Nevertheless, it is an important issue in dependable systems. Combined with the broad section of other research [32, 41, 25], this work provides further evidence that event-triggered communication has a genuine and important role in dependable real-time communication.

## References


[1] ARINC. *ARINC Specification 629-4*. Aeronautical Radio Ltd, Apr 1996.

[2] I. J. Bate. *Scheduling and Timing Analysis for Safety Critical Real-Time Systems*. PhD thesis, Department of Computer Science, University of York, York, YO10 5DD, 1999.

[3] R. Belschner, J. Berwanger, C. Ebner, H. Eisele, S. Führer, T. Forest, T. Führer, F. Hartwich, B. Hedenetz, R. Hugel, A. Knapp, J. Krammer, A. Millsap, B. Müller, M. Peller, and A. Schedl. *FlexRay Requirements Specification*. FlexRay, www.flexray.com, v2.0.2 edition, Apr 2002.

[4] Bosch, Postfach 50, D-700 Stuttgart 1. *CAN Specification*, version 2.0 edition, 1991.

[5] I. Broster. *Flexibility in Dependable Communication*. PhD thesis, Department of Computer Science, University of York, York, YO10 5DD, UK, Aug 2003.

[6] I. Broster, G. Bernat, and A. Burns. Weakly hard constraints on Controller Area Network. In *Proceedings of 14th Euromicro Conference on Real-time Systems*, Vienna, Austria, Jun 2002. IEEE.

[7] I. Broster and A. Burns. Timely use of the CAN protocol in critical hard real-time systems with faults. In *Euromicro Conference on Real-time Systems*, pages 95–102, Delft, The Netherlands, Jun 2001. IEEE Computer Society.

[8] I. Broster and A. Burns. An analysable bus-guardian for event-triggered communication. In *Proceedings of the 24th Real-time Systems Symposium*, pages 410–419, Cancun, Mexico, Dec 2003. Computer Society, IEEE.

[9] I. Broster, A. Burns, and G. Rodríguez-Navas. Probabilistic analysis of can with faults. In *Proceedings of the 23rd Real-time Systems Symposium*, Austin, Texas, Dec 2002. IEEE.

[10] M. J. Buckingham. *Noise in Electronic Devices and Systems*. Series in Electrical and Electronic Engineering. Ellis Horwood/Wiley, 1983.

[11] A. Burns. Real-time systems. In *Encyclopedia of Physical Science and Technology*, volume 14, pages 45–54. Academic Press, 2002.

[12] A. Burns, S. Punnekkat, L. Strigini, and D. Wright. Probabilistic scheduling guarantees for fault-tolerant real-time systems. Technical Report YCS-311, Department of Computer Science, University of York, 1998.

[13] J. Ferreira, L. Almeida, E. Martins, P. Pedreiras, and J. Fonseca. Components to enforce fail-silence behavior in dynamic master-slave systems. In *Proceedings SICICA'2003 - 5th IFAC International Symposium on Intelligent Components and Instruments for Control Applications*, Aveiro, Portugal, Jul 2003.

[14] T. Führer, B. Muller, W. Dieterle, F. Hartwich, R. Hugel, and M. Walther. Time triggered communication on CAN. Technical report, Robert Bosch GmbH, 2000. Available from `http://www.can.bosch.com/`.

[15] F. Hartwich, B. Muller, T. Führer, and R. Hugel. CAN network with time-triggered communication. In *7th international CAN Conference*. CAN in Automation GmbH, Oct 2000.

[16] A. Helfrick. Avionics & portable electronics : Trouble in the air? *Avionics News Magazine*, Sept 1996.

[17] IEE. EMC and functional safety. IEE guidance document, IEE, Sept 2000. Available from `http://www.iee.org.uk/PAB/EMC/core.htm`.

[18] International Standards Organisation. *ISO 11898. Road Vehicles—Interchange of digital information—Controller area network (CAN) for high speed communication*, 1993.

[19] H. Kim and K. G. Shin. Modeling of externally-induced/common-cause faults in fault-tolerant systems. Technical report, Real-time Computing Lab, Department of Electrical Engineering and Computer Science, University of Michigan, 1993.

[20] H. Kopetz. A solution to an automotive control system benchmark. In *Proc. 15th IEEE Real-Time Systems Symposium*, pages 154–158, Puerto Rico, Dec 1994. IEEE.

[21] H. Kopetz. *Real-Time Systems: Design Principles for Distributed Embedded Applications*. Kluwer Academic, 1997.

[22] H. Kopetz. A comparison of CAN and TTP. Technical Report 1998, Technishe Universitat Wien, Austria, 1998.

[23] H. Kopetz. Time-triggered model of computation. In *Proceedings 19th Real-Time Systems Symposium*, pages 168–177, Madrid, Spain, Dec 1998.

[24] P. B. Ladkin. Electromagnetic interference with aircraft systems: why worry? Technical Report RVS-J-97-03, University of Bielefeld—Faculty of Technology, 1997.

[25] G. M. A. Lima. *Fault Tolerance in Fixed-Priority Hard Real-Time Distributed Systems*. PhD thesis, Department of Computer Science, University of York, York, YO10 5DD, UK, May 2003.

[26] G. M. A. Lima and A. Burns. A consensus protocol for CAN-based systems. In *Proceedings of the 24th Real-time Systems Symposium*, pages 420–429, Cancun, Mexico, Dec 2003. Computer Society, IEEE.

[27] N. Navet, Y.-Q.Song, and F. Simonot. Worst-case deadline failure probability in real-time applications distributed over controller area network. *Journal of Systems Architecture*, 46(1):607–617, 2000.

[28] T. Nolte, H. Hansson, C. Norström, and S. Punnekkat. Using bit-stuffing distributions in CAN analysis. In *IEEE Real-Time Embedded Systems Workshop at the Real-Time Systems Symposium*, London, UK, 2001.

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

[30] J. Proenza and J. Miro-Julia. MajorCAN: A modification to the Controller Area Network protocol to achieve Atomic Broadcast. In *IEEE Int. Workshop on Group Communications and Computations (IWGCC)*, Taipei, Taiwan, Apr 2000.

[31] S. Punnekkat, H. Hansson, and C. Norström. Response time analysis under errors for CAN. In *Proceedings of the 6th Real-Time Technology and Applications Symposium (RTAS)*, pages 258–265, Washington DC, 2000. IEEE.

[32] J. Rufino. *Computational System for Real-time Distributed Control*. PhD thesis, Universidade Técnica de Lisboa Instito Superior Técnico, Jul 2002.

[33] J. Rufino, P. Veríssimo, and G. Arroz. Embedded platforms for distributed real-time computing: Challenges and results. In *Proceedings of the 2nd International Symposium on Object-oriented Real-time distributed Computing*, pages 147–152, Saint Malo, France, May 1999. IEEE.

[34] SAE. Class C application requirement considerations. Technical Report J2056/1, Society of Automotive Engineers, 1993.

[35] N. J. A. Sloane. Sequence a000108—catalan numbers. Available from `http://www.research.att.com/~njas/sequences/Seis.html`, 2003. On-Line Encyclopedia of Integer Sequences.

[36] C. Temple. Avoiding the babbling-idiot failure in a time-triggered communication system. In *Proceedings 28th Annual International Fault Tolerant Computing Symposium, FTCS'98*, 1998.

[37] K. Tindell and A. Burns. Guaranteed message latencies for distributed safety-critical hard real-time networks. Technical Report YCS 229, Department of Computer Science, University of York, May 1994.

[38] K. Tindell, A. Burns, and A. J. Wellings. Calculating controller area network (CAN) message response times. *Control Engineering Practice*, 3(8):1163–1169, 1995.

[39] K. Tindell and H. Hansson. Babbling idiots, the dual-priority protocol, and smart CAN controllers. In *Proceedings of the 2nd International CAN Conference*, pages 7.22–28, 1995.

[40] TTP. TTP/C protocol specification. Technical report, TT-Tech Computertechnik, Wien, Austria, Jul 1999.

[41] 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*, pages 112–121, Seatle, Washington, USA, Jun 1997. IEEE.