Fault-Tolerance Analysis of Mixed CAN/Switched Ethernet Architecture

Cláudia Betous-Almeida, Jean-Luc Scharbarg,
Christian Fraboul

IRIT-ENSEEIHT-University of Toulouse
2, Rue Charles Camichel
31000 Toulouse, France
claudia.betous,jean-luc.scharbarg, christian.fraboul@enseeiht.fr

Abstract. CAN is a well known fieldbus standard used in safety critical applications of embedded systems. However, steadily increasing amount of exchanged information in such systems has led to the use of Switched Ethernet like solutions. Mixed CAN/Switched Ethernet architectures allow to bypass CAN limitations while preserving the widely used CAN technology. In order to use this kind of architecture in safety critical applications a complete fault tolerance analysis is mandatory. In this paper, we use a simulation-based fault-injection technique to analyse the impact of different types of errors on the percentage of application frames missing their deadlines. Results show that different types of errors don't have the same impact on different types of traffic. Moreover, it is shown that the re-emission of corrupted frames can have a negative impact on the system's global performance.

1 Introduction

The Controller Area Network (ISO-CAN, 1993) is a well-known fieldbus standard that provides a real-time performance with a fair reliability degree, at a very low cost. The growing use of CAN in safety-critical real-time applications of embedded systems, such as automotive or avionic ones, has led to concerns regarding the reliability evaluation of these systems. Moreover, the amount of exchanged information in such systems has steadily increased over the years and is now reaching the traditional fieldbusses' limits, namely bandwidth limits.

So as to overcome those limits, Switched Ethernet like solutions are more and more envisioned, for example in avionics systems with the AFDX (ARI, 2002), (Charara et al., 2006). We have proposed a mixed CAN/Switched Ethernet architecture as an alternative between pure CAN and pure Switched Ethernet architectures (Scharbarg et al., 2005b).

In order to successfully use this architecture in safety-critical applications, a complete fault-tolerance analysis is needed. Fault-injection is the technique the most often used by system's designers in order to analyse the dynamic behaviour of the system, in the presence of faults.

This paper is an elaboration of the work presented in Betous-Almeida et al. (2006). In this paper we present an overview of the different types of models and consequently different fault-

models that can be used in a simulation-based fault-injection technique. Namely, the results presented include more elaborated fault-models and different injection locations.

The remainder of the paper is organised as follows. Section 2 presents the CAN/Switched Ethernet architecture used in this study. In Section 3 we discuss the different parameters: type of model, type of fault-model and description level. Section 4 describes our experimental approach and gives results obtained with the experiments. Finally, Section 5 summarises the work and suggests future considerations.

2 Mixed CAN/Switched Ethernet Architecture

The network architecture includes the two communication technologies CAN and Switched Ethernet. In this section, we present briefly those two technologies. Then, we describe the network architecture that we will consider in the remaining of the paper. Finally, we explain the kinds of traffic considered over the network architecture.

2.1 CAN Protocol

CAN (Controller Area Network) ISO-CAN (1993) is a serial communication protocol suited for networking sensors, actuators and other nodes in real-time systems. The CAN specification defines several versions of the protocol for the physical and the data link layer. In this paper, we shortly present CAN 2.0 A.

The CAN addressing system is based on message identifiers: a frame does not have a destination nor a source address. All frames are broadcasted on the bus. Stations get the frames they are interested in by a filtering process of the identifiers.

The frame format is depicted in Fig. 1. The relevant fields for the remainder of the paper are: the identifier field (which identifies the data contained in the frame), the DLC field (which gives the length, in bytes, of the data field) and the data field which is the payload of the frame.

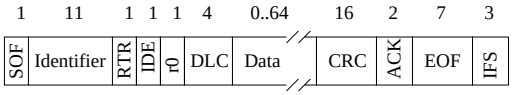

FIG. 1 – *CAN frame (sizes in bits).*

Bit-stuffing is used to avoid the transmission of long sequences of bits with identical value (Nolte et al., 2001). As soon as 5 bits of identical value are transmitted, a bit of opposite value is automatically inserted. This mechanism is valid for the whole frame, except IFS, EOF, ACK and the last bit of CRC.

The medium access method (MAC) is CSMA/CR (Carrier Sense Multiple Access / Collision Resolution): the starting of frame transmissions on the bus are synchronous. When two or more stations start a transmission simultaneously, the one with the highest priority identifier (lowest value) wins and the others stop their transmission. This is implemented by a collision detection on a bit by bit basis. When a station transmits 1 (recessive bit) and detects 0 (dominant bit), it knows that a frame with a higher priority is being transmitted and, consequently, it

immediately stops transmission. This mechanism guaranties strict priority order on identifiers, provided identifiers are unique. It implies limitations of the bandwidth and the maximal length of the bus (e.g. 1 Mbs for 40 meters).

A data frame is acknowledged using the first bit of the ACK field. It is transmitted recessive by the emitter of the frame and dominant by each station that receives successfully the frame. That means that, when the emitter detects a recessive value on the first bit of the ACK field, it can conclude that no station has received its frame correctly.

CAN offers powerful error detection mechanisms. As soon as a station detects an error on the bus, it emits an error frame, composed of an error flag (six dominant bits), followed by an error delimiter (eight recessive bits). As every station has to detect the error, the length of the aggregate error frames is between 14 and 20 bits (see ISO-CAN (1993) for details). The aggregate error frame is followed by an interframe space and another data frame transmission. The following error types are detected:

1. 6 consecutive bits with the same value during the frame portion where bit-stuffing is active;
2. a dominant bit is transmitted and a recessive bit is received;
3. a recessive bit is transmitted outside the Identifier and ACK fields and a dominant bit is received;
4. the received and computed CRC are different;
5. the first bit of the ACK field is received recessive;
6. a fixed bit is received with a wrong value (e.g. the last bit of the CRC field is received dominant).

Also, in order to limit the consequences of a permanently faulty station on the whole network, a confining mechanism is used (see ISO-CAN (1993) for details).

2.2 Full Duplex Switched Ethernet

Full duplex switched Ethernet is an enhancement of Ethernet. The Ethernet link layer (IEEE802.3, 2002) is designed for computer local networks where high bandwidth and low cost hardware is more important than guaranteed deadlines and/or jitter.

The Ethernet addressing system is based on MAC addresses: each Ethernet entity has a unique MAC address. In each frame, the destination (unicast, broadcast or multicast) and source addresses are inserted. Frames are broadcasted on the physical layer. Entities get the frames there are interested in by a filtering process.

Full duplex switched Ethernet is a way to bypass the CSMA/CD medium access strategy of Ethernet: each station is directly connected to an Ethernet switch with a full duplex link. This way the medium is always free. Consequently guaranteed performances are strongly connected to policies of the switch. Many literature has been devoted to the subject (see for instance Zhang (1995) concerning service disciplines in packet-switching networks). In this paper, we consider a very basic switch with a First-In First-Out policy on each output port.

2.3 Heterogeneous CAN/Switched Ethernet Architectures

The network architecture considered in this paper interconnects several CAN busses with a full duplex Switched Ethernet network. An example of such an architecture is depicted in

Fig. 2. It includes four CAN busses and one Ethernet switch. There is a bridge station between each CAN bus and the switch. The switch has four receive ports and four queued transmit ports. When a frame arrives at the switch, the control logic determines the transmit port and tries to transmit the frame immediately. If the port is busy because another frame is already being sent, the frame is stored in the first-in first-out transmit port queue. The memory to store pending frames is obtained from a shared memory pool. If no more memory is available, the received frame is dropped.

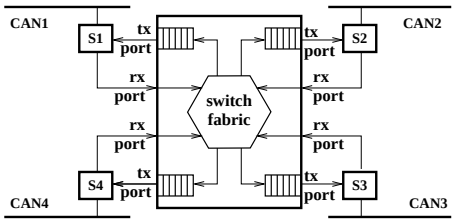

FIG. 2 – *Network architecture.*

More generally, the architecture includes N_c CAN busses and the switch has N_c receive ports and N_c queued transmit ports. Network architectures with more than one switch are not considered in this paper. Moreover, there won't be any non-CAN stations connected to the Ethernet switch.

2.4 Application Traffic Over the Network

The traffic on the whole network can be divided in two kinds:

1. local CAN traffic: all the frames of this traffic are produced by a station on a CAN data bus s and consumed by stations all on the same CAN data bus s; consequently, those frames don't have to be transmitted on the Switched Ethernet;
2. global CAN traffic: all the frames of this traffic are produced by a station on a CAN data bus s (their home bus) and consumed by stations among which at least one is on a CAN data bus d with $d \neq s$ (all those d busses are called the distant busses of the global CAN frame); consequently, those frames have to be transmitted on Switched Ethernet.

CAN traffic (local and global) is composed of messages. Each message M_i consists in the periodic production of a frame with a given DLC. Message M_i period is denoted P_i. Each frame of M_i has a relative deadline equal to the period P_i. A global message is not transferred on a CAN bus which is neither its home bus nor one of its distant busses.

Concerning the scheduling of frames on CAN data busses, identifiers are allocated to CAN messages following a rate monotonic policy (Liu and Layland, 1973).

As global CAN traffic has to be transmitted on the Switched Ethernet network, it is necessary to define a bridging strategy between CAN and Switched Ethernet. As explained in Scharbarg et al. (2005a), the very different CAN and Ethernet characteristics make an encapsulating policy the best choice. The encapsulation consists in putting the Identifier, DLC and Data fields of CAN frames in the Data field of the Ethernet frame (the other fields of CAN

frames can be easily reconstructed). This means that a CAN frame occupies at most 10 bytes of the Data field of an Ethernet frame. Different strategies have been compared by simulation in a vintage Ethernet context in Scharbarg et al. (2005a) for pure CAN traffic and Scharbarg et al. (2005b) considering additional non-CAN traffic.

In this paper, we consider the one for one strategy (each CAN frame is put in a separate Ethernet frame and transmitted as soon as possible).

3 Fault-Tolerance Analysis

The fault-tolerance analysis of a systems allows the validation of the error detection mechanisms. Among all the techniques that can be utilised to do a fault-tolerance analysis, fault-injection techniques are the most frequently applied on fieldbusses by system designers.

Fault-injection is the deliberate introduction of faults into a system (Arlat et al., 1990). It can be seen as a technique for testing a fault-tolerant system in respect to a class of inputs specific to such a system, meaning faults.

In Betous-Almeida et al. (2006) we have analysed the three main categories of fault-injection techniques: simulation-based, hardware fault-injection and software fault-injection. In this paper we consider a simulation-based approach.

In the next section we will discuss the choices for the type of models and fault-models that can be used in the experiments.

3.1 Simulation-Based Fault-Injection

Simulation-based fault-injection is a fault-injection technique that is mainly used in the early phases of a system's development. It allows us to obtain a first draft of results namely in the analysis and measurement of error propagation in simulation models.

This fault-injection technique implies the use of a system's model to be used in the experiments. In Carter and Abraham (1987) the authors distinguish three major types of models:

1. *axiomatic* models, for instance analytical models which model the structure and the dependability and/or performance evaluation of the system. Examples of these models are: reliability block diagrams, fault trees, Markov models or stochastic Petri nets;
2. *empirical* models, these models take into account more detailed structural and behavioural descriptions that require a simulation to process them. Note that it is not the construction of this type of models that is empirical but rather its process;
3. *physical* models, prototypes implementing the hardware and/or the software features of the system.

Depending on the chosen model, the fault-model utilised in the experiments is different. Indeed, as said in Arlat et al. (1990), the choice of an axiomatic model implies the use of stochastic processes (mostly Poisson processes) as faults. Whereas the use of empiric models enable the use of more realistic distributions as it allows the use of different description levels. Finally, in the case of physical models, the set of faults considered is mainly based on physical faults.

In the next section we summarise the main different types of fault-models frequently used in fault-injection.

3.2 Fault-Models

A fault-model is a set of faults of the same nature, i.e. they produce the same type of reaction in the system's behaviour.

As said before, the choice of the fault-model depends on the type of model used, but it also depends on the type of faults we want to inject (DBench, 2002). Depending on their duration, examples of the different fault-models are:

1. transient: faults which usually have a short temporal duration (e.g. *bit-flip*, *pulse*, *delay* or *indetermination* fault-models);
2. permanent: faults which remain in existence indefinitely, (e.g. *stuck-at*, *stuck-open*, *bridging* fault-model);
3. intermittent: faults that have a temporal duration, but unlike transient they appear and disappear repeatedly in time, without a periodical behaviour. Due to their characteristics, fault-models applicable to this type of faults are the same as those used in permanent faults.

The choice of the fault-model depends also on the description level: component, gate, circuit or system-level.

3.3 Injection Location

The last parameter to be considered is the fault-injection location. Considering our architecture, we have identified five possible injection locations which, when considering a global CAN traffic, are: CAN data bus sender and receiver, Switched Ethernet, Switched Ethernet receive port and queued transmit port.

In a previous paper (Betous-Almeida et al., 2006) we have analysed the effects of single faults injected on a unique location: the CAN data bus sender. For this paper, we have studied the effects of faults injected in the CAN data bus sender and receiver, but also in the Switched Ethernet queued transmit port. Moreover, we have analysed the network's behaviour when single errors are detected by the embedded mechanisms but also when a burst of errors[1] occurs.

4 Experimental Results

In this section we describe our experimental approach and summarise some results concerning the fault-injection experiments we have done so far.

4.1 Example Application

The dependability analysis will be conducted using the example application depicted in Table 1. It includes 12 periodic messages, transmitted over a network architecture similar to the one depicted in Figure 2. The relative deadline of each message is equal to its period. The values for length and transmission time correspond to a 1 Mbs CAN bus. Let's have a look at the first line. It means there are 8 periodic local CAN messages of period 4 ms for CAN bus 1. Those messages will be called M_1 messages in the following. Each occurrence of an M_1 message

[1] A burst of errors is a sequence of single errors.

Type	Kind mes.	Per. (bits)	Data (bits)	Lg.
M_1	Local C1	4	32	95
M_1	C3 $\Rightarrow$ C4	4	32	95
M_2	Local C4	4	16	75
M_2	C2 $\Rightarrow$ C3	4	16	75
M_3	Local C2	4	32	95
M_3	C1 $\Rightarrow$ C2	4	32	95
M_4	Local C3	4	16	75
M_4	C2 $\Rightarrow$ C3	4	16	75
M_5	Local C2	10	32	95
M_5	C2 $\Rightarrow$ C3	10	32	95
M_6	Local C4	10	16	75
M_6	C4 $\Rightarrow$ C1	10	16	75

TAB. 1 – *Message sets of the application.*

contains 8 bytes of data. The length of an occurrence of the message is 135 bits. It is computed using the following formula:

$$length = 47 + 8 \times DLC + \left\lfloor \frac{34 + 8 \times DLC}{4} \right\rfloor \quad (1)$$

47 is the number of control bits of a CAN frame, including the interframe space. $8 \times DLC$ is the number of data bits of the frame. The remainder of the formula is the maximum number of stuff bits inserted in the frame. So, it is a worst case length. The second line of Table 1 concerns global CAN messages with similar characteristics. They are produced by a station on CAN bus 1 and consumed by at least one station on CAN bus 2.

4.2 Description of the Experiments

For the purpose of the fault-injection experiments we conducted, we have developed a model of the mixed CAN/Switched Ethernet network using QNAP2 from INRIA/Simulog (Simulog), which is a software tool allowing the modelling simulation and analysis of queueing networks.

During fault injection experiments, the occurrence rate of frames timeout, or in other words, not transmitted once the given time limit for transmission was reached, is recorded and stored for analysis. Two types of frames are considered, local and global: the difference between the two types being that local frames do not transit by the switch, global ones do (see section 2.4 for details).

In order to model the burst errors, we used a model based on the widely used *Gilbert-Elliot* model (Gilbert (1960), Elliot (1963)). We have consider a two state Markov chain, see Figure 3. To each state we have assigned a constant bit error rate.

In these experiments we consider that the “ok” state has a 0% error rate and that the “ko” state has a 80% error rate.

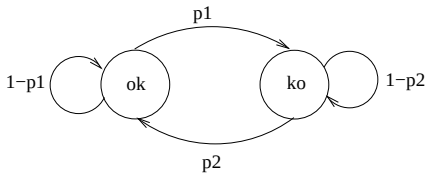

FIG. 3 – *Two state Markov chain.*

Figure	CAN error rate		Switch error rate
	Burst BER	Single BER	
Fig. 4	80%	0%	0%
Fig. 5	80%	15%	15%
Fig. 7	80%	15%	0%
Fig. 6	80%	0%	15%

TAB. 2 – *Parameters values of presented experiments results.*

Moreover, the fault-injection experiment conditions, for the results presented in this paper, are :

- *Fault injection location:* CAN bus sender, CAN bus receiver, Ethernet switch;
- *Fault model:* bit-flip, in single and burst errors;
- *Injection instant:* randomly selected during the frame transmission.

We have classified the effects of faults according to two categories:

- *Timeout:* frames missed their deadlines;
- *Performance degradation:* some of the available bandwidth is wasted.

4.3 Effects of Faults

In this section we present some of the results obtained after different fault-injection campaigns.

The goal of our first campaign was to analyse the system's behaviour in the presence of burst errors. The burst error probability p is implemented as follows: time is decomposed in intervals of x ms. p is the probability that each interval suffers a burst of bit-flip errors. As said before, the bit error rate in this case will be of 80%.

Table 2 summarises the values of the parameters considered in the different fault-injection campaigns. In all of the shown results, p={0, 0.02, 0.04, 0.06, 0.08, 0.10, 0.12, 0.14, 0.16, 0.18, 0.20}.

All the results show that the percentage of CAN frames (local and global) missing their deadlines grow linearly with p (burst probability). It means that errors occurring during a burst interval have only little influence on following no burst intervals. It means that re-emission of corrupted frames disturbs other frames only during a short time after the burst interval.

Figure 4 shows the percentage of local and global CAN frames missing their deadlines given a burst of bit-flip errors probability on the CAN busses senders and receivers. Moreover, in this case there are no Ethernet switch errors and no single errors. We can see that the number of global CAN frames missing their deadlines increases slightly faster than the local ones. This can be explained by the fact that the laxity for the global frames is smaller than the laxity of the local ones. Also, since global frames transit over at least two CAN busses the probability for them to suffer a burst error is higher than local ones.

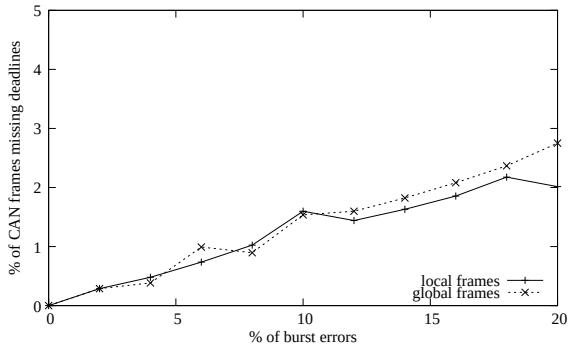

FIG. 4 – *Missed deadlines: no switch errors, no single error.*

In Figure 5 there are two interesting results: the fact that single errors are taken into account, increases the number of global and local frames missing their deadlines. The reason for it is that both types of frames suffer from bit errors. Secondly, the introduction of switch errors induces a uniform increase of number of global frames missing their deadlines. This is due to the fact that this kind of error has no impact on local frames.

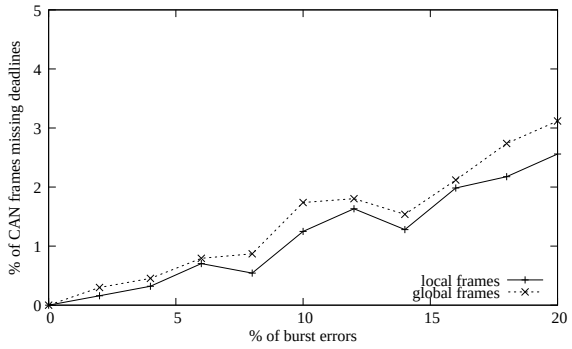

FIG. 5 – *Missed deadlines: 15% switch errors, 15% probability of single error.*

In Figure 7, we can see that the number of frames missing their deadlines is higher regarding both local and global frames. This is due to the fact that single errors affect both types of

frames. Since the percentage of these errors is higher, it is logical that the number of frames missing their deadlines is higher.

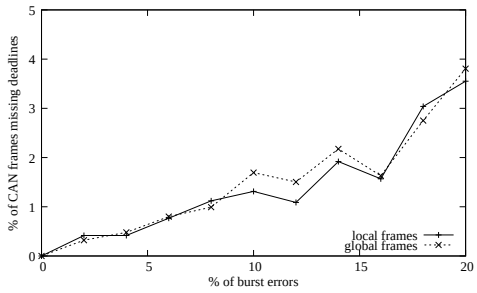

FIG. 6 – *Missed deadlines: 15% probability of single error, 0% switch errors.*

Considering Figure 6, two results are worth notice: we see that the number of local frames missing their deadlines is not much different from that of Figure 5. However, the number of global frames missing their deadlines is higher, specially with high burst error rates. This is not surprising since we have the same rate for single errors but a higher probability in errors in the Ethernet switch. Note that these errors affect only global frames.

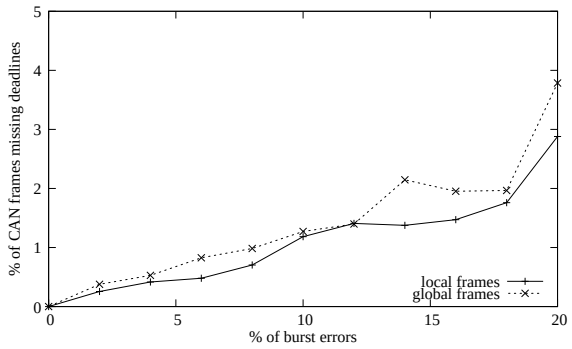

FIG. 7 – *Missed deadlines: 0% probability of single error, 15% switch errors.*

Figure 8 shows the impact of frame re-emission due to error on the global performance of the system. More precisely, we determine the difference between the total number of frames (local and global) that miss their deadlines and the number of frames in error (which is equal to the number of frames re-emitted). We observe that this difference is always negative which means that frame re-emission, with the current workload, does not degrades the global performance of the system. Also, we notice that the number of re-emissions is higher with just single errors than with just switch errors. This is logic since single errors affect both types of traffic.

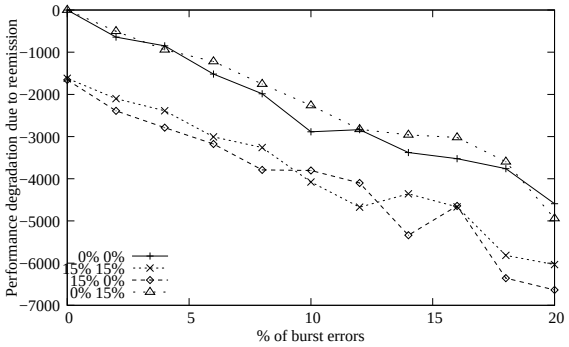

FIG. 8 – *Performance degradation.*

5 Conclusions and Future Work

In this paper, we have presented a fault-tolerance analysis of a communication network using a mixed CAN/Switched Ethernet architecture, made of several CAN busses interconnected by an Ethernet switch.

We have considered a pure CAN periodic traffic composed of local frames that do not transit on the Ethernet, and global frames that are each encapsulated in a separate Ethernet frame.

Concerning the fault-tolerance study, an overview of the simulation-based fault injection technique was made. We presented the different types of models that can be used, and the different types of fault-models We have laid out the conditions used in the simulations, namely the use of the *bit-flip* fault model as single errors but also as burst errors on different locations: CAN bus sender, receiver and in the Ethernet switch.

We have shown that the number of frames missing their deadlines is linearly related to the bit error's probability. Results have shown that different types of errors don't have the same impact on different types of traffic.

In the near future we will continue carrying out fault injection experiments in the mixed CAN/Switched Ethernet network. The first series of experiments will concern changes in the workload in order to study more precisely the impact of frame re-emission.

The study presented in this paper considers a theoretical traffic, we intend to apply this method on an avionics context (Charara et al., 2006). For this type of application, it will be necessary to evaluate other CAN/Ethernet encapsulation strategies (Scharbarg et al., 2005b), in the fault injection context. Moreover, it will be mandatory to study the global behaviour of the system when adding non CAN traffic over Ethernet. It will also be necessary to consider an Ethernet switch with a more sophisticated service policy.

References

(2002). *ARINC 664, Aircraft Data Network, Part 1: Systems Concepts and Overview.*

Arlat, J., M. Aguera, L. Amat, Y. Crouzet, J.-C. Fabre, J.-C. Laprie, E. Martins, and D. Powell (1990). Fault injection for dependability validation : a methodology and some applications. *IEEE Transactions on Software Engineering 16*(2), 166–182.

Betous-Almeida, C., J.-L. Scharbarg, and C. Fraboul (2006). Dependability analysis of mixed can/switched ethernet network. In *Proceedings of ESREL 2006*, Volume 3, pp. 2249–2256.

Carter, W. C. and J. Abraham (1987). Design and evaluation tools for fault-tolerant systems. In *Proceedings of AIAA Computers in Aerospace Conference*, pp. 70–77.

Charara, H., J.-L. Scharbarg, J. Ermont, and C. Fraboul (2006). Methods for bounding end-to-end delays on an afdx network. In *Proceedings of the 18th ECRTS*, Germany.

DBench (2002). Fault representativeness. Technical Report online: http://www.laas.fr/DBench/ETIE2.pdf, Deliverable ETIE2 of the Dependability Benchmarking (DBench) Project.

Elliot, E. (1963). Estimates of error rates for codes on burst-noise channels. *Bell Syst. Tech. J. 45*(2), 1977–1997.

Gilbert, E. (1960). Capacity of a burst-noise channel. *Bell Syst. Tech. J. 39*, 1253–1265.

IEEE802.3 (2002). CSMA/CD access method. IEEE Standard 802.3, IEEE.

ISO-CAN (1993). Road vehicles – controller area network (can). ISO Standard 11898, International Organization for Standarization.

Liu, C. and J. Layland (1973). Scheduling algorithms for multiprogramming in hard real-time environment. *Journal of ACM 20*(1), 46–61.

Nolte, T., H. Hansson, C. Norstrom, and S. Punnekat (2001). Using bit-stuffing distributions in CAN analysis. In *IEEE/IEE Real-Time Embedded Systems Workshop (RTES)*.

Scharbarg, J.-L., M. Boyer, and C. Fraboul (2005a). Can-ethernet architectures for real-time applications. In *Proc. of the 10th International Conference on Emerging Technologies and Factory Automation*, Italy. IEEE.

Scharbarg, J.-L., M. Boyer, and C. Fraboul (2005b). Interconnecting can busses via an ethernet backbone. In *Proc. of the 6th International Conference on Fieldbus Systems and their Applications*, Mexico. IFAC.

Simulog. Qnap2. http://www.simulog.fr.

Zhang, H. (1995). Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE 83*(10), 1374–1396.