Schedulability Guarantees for Dependable Distributed Real-Time Systems under Error Bursts

Huseyin Aysan, Radu Dobrin, and Sasikumar Punnekkat

Mälardalen University, Västerås, Sweden
{huseyin.aysan,radu.dobrin,sasikumar.punnekkat}@mdh.se

Abstract. In dependable embedded real-time systems, typically built of computing nodes exchanging messages over reliability-constrained networks, the provision of schedulability guarantees for task and message sets under realistic fault and error assumptions is an essential requirement, though complex and tricky to achieve. An important factor to be considered in this context is the random nature of occurrences of faults and errors, which, if addressed in the traditional schedulability analysis by assuming a rigid worst-case occurrence scenario, may lead to inaccurate results. In this work we propose a framework for end-to-end probabilistic schedulability analysis for real-time tasks exchanging messages over Controller Area Network under stochastic errors.

Keywords: real-time systems, task scheduling, CAN, dependability, reliability, fault tolerance, schedulability analysis.

1 Introduction

Real-time systems are computer systems in which the correctness of the system depends not only on the logical correctness of the computations performed, but also on which point in time the results are provided [1]. Delivering a result at a point in time beyond the latest possible, i.e., after its deadline, may result to catastrophic consequences in *safety critical real-time systems*. Examples of such systems are medical control equipment nuclear power plants, or vehicle control systems.

A real-time system typically consist of a number of a number of *resources* (e.g., processing nodes connected by communication mediums), a number of *tasks* typically communicating over a field buss, e.g., Controller Area Network (CAN) by exchanging *messages*, designed to fulfil a number of *timing constraints*, and a *scheduler* that assigns each task and message a fraction of the processor(s) time or bus bandwidth, according to a *scheduling policy*. Events like tasks or messages are usually *periodic* or *non-periodic* depending on their pattern of occurrences. While periodic events consist of an infinite sequence of invocations, called *instances*, non-periodic ones are invoked by the occurrence of another event. The choice of

I. Dumitrache (Ed.): Adv. in Intelligent Control Systems & Computer Science, AISC 187, pp. 393–406.
springerlink.com

tasks, messages and scheduling policy is made by the system designer to satisfy some original constraints imposed on the system. Consequently, tasks and messages are assigned a number of scheduling parameters, such as periods, deadlines or priorities, depending on the chosen scheduling policy.

Besides the real-time specific deadline constraint, the majority of safety critical real-time systems are typically characterized by dependability requirements. In essence, these systems have the major design objective to guarantee the properties of correctness and timeliness even under error occurrences. Further, these systems are typically built using several computing nodes that interact with each other in a distributed manner where reliable communication plays a crucial role for achieving overall system dependability. While the deadline constraint is addressed by using the response time analysis, the design of reliable end-to-end systems, involving task executing on processing nodes and exchanging messages over a network (i.e. CAN), requires usage of appropriate fault-tolerance (FT) mechanisms and analysis techniques jointly at node- as well as network level. However, the fundamental requirement for the design of effective and efficient FT mechanisms is a realistic and applicable model of potential faults, their manifestations and consequences.

In a large number of safety or mission critical systems, that typically employ the *preemptive* fixed priority scheduling (FPS) policy, real-time schedulability analysis techniques have been increasingly used in order to ensure that the strict timeliness requirements of the applications are met. The preemptive behaviour of FPS, although desirable to increase the schedulability bound, can on the other hand benefit of control mechanisms to address the potential high contexts switch costs [2]. The analysis has been also extended to CAN, where real-time messages are scheduled *non-preemptively* on the bus. CAN was designed in the 1980s at Robert Bosch GmbH [3] with a special focus on automotive real-time requirements and has been widely used in the automotive and automation industries due to its ease in use, low cost and provided reduction in wiring complexity. The most important feature of CAN from the real-time perspective is its predictable behaviour. Recent works have addressed the effect of the network delay on the performance of control systems [4]. CAN provides means for prioritized control of the transmission medium by using an arbitration mechanism which guarantees that the highest priority message that enters an arbitration will be transmitted first. This makes CAN amenable to response time analysis akin to those performed on fixed priority task sets. Volcano methodology used by Volvo [5] is an example of the acceptance of such analysis by the industry. The model underlying the basic CAN analysis assumes an error free communication bus, i.e. all messages sent are assumed to be correctly received, which may not always be true due to the interference from the operational environment or the faulty hardware components. These interferences cause errors in the transmitted data, which could indirectly lead to catastrophic failures. While in processor scheduling the designer is responsible to provide fault tolerance mechanisms, in CAN scheduling, to reduce the risks due to erroneous transmissions, CAN designers have provided elaborate error checking and error confinement features in the protocol. The basic philosophy of these features is to identify an error as fast as possible and then retransmit the affected message. This implies that in systems without spatial redundancy of communication medium/controllers, the FT mechanism employed is time redundancy which addresses transient errors but could have an adverse impact on the latencies of message sets; potentially leading to violation of timing requirements. Furthermore, burst of errors typically affect several message retransmission attempts and contribute to potentially large response time that may deem the system unschedulable.

In this work, we propose an end-to-end schedulability analysis for real-time tasks executing on processing nodes and exchanging messages over CAN, under error scenarios.

2 End-to-End Probabilistic Fault-Tolerance Analysis (PFTA) under Error Bursts

In this section we present an end-to-end probabilistic fault-tolerance analysis (PFTA) for distributed real-time systems under error bursts. We assume a number of processing nodes executing real-time tasks that exchange messages over CAN. We first present PFTA for a single processor node where tasks execute scheduled under FPS. Then we show how PFTA is performed for message scheduling in CAN, and finally we introduce PFTA for transactions of tasks exchanging messages over CAN under error bursts.

2.1 PFTA for Processor Scheduling under Error Bursts

The approach begins with a performing a set of schedulability analyses that accounts for a range of worst-case scenarios generated by stochastic error burst occurrences on the response times of tasks scheduled under the fixed priority scheduling (FPS) policy. Then the probabilistic schedulability guarantees are calculated as a weighted sum of the conditional probabilities of schedulability under specified error burst characteristics.

In this subsection a single processor platform is assumed on which a sporadic task set is allocated which have deadlines equal to or less than their minimum inter-arrival times. Whenever an error is detected within a task, the affected task τ_i executes an alternate task with a worst-case execution time less than or equal to the original worst-case execution time of its primary, a deadline equal to the original deadline and a minimum inter-arrival time equal to the minimum inter-arrival-time of its primary. This alternate can typically be a re-execution of the same task, a recovery block, an exception handler, or an alternate task with imprecise computations. Errors are assumed to be detected just before the completion of the affected task instances.

The main sources of errors are assumed to be electromagnetic interferences (external faults), and transient hardware faults (internal faults) that affect, e.g. the sensors and the network systems. Examples to the considered errors are incorrect input values from sensors, or failure in delivering the output values via network messages. Errors that are propagated into tasks are detected at the end of task executions by observing, e.g., the out-of-range output values or omitted outputs.

Examples to the assumed error detection mechanisms are usage of sanity checks, range checks, checksums for the value correctness and the usage of watchdog timers for the time correctness. Watchdog timers are assumed to be implemented as simple hardware units that run in parallel with the tasks and interrupt in case of detected errors and the overhead of the value error detectors are included in the WCETs of the respective tasks.

Each fault may result in errors in the form of an error burst for a random duration. The distribution regarding the duration of the faults is very much domain specific, and, in this section, it is assumed that the information regarding the probability distribution of the fault durations is available. Other parameters assumed to be given related to the error model is the error rate λ and the mission time L.

In this work, it is assumed that no task can successfully finish between errors within a burst. Furthermore, any task instance scheduled even partially under the error burst will be considered as affected by the error.

Methodology Overview. The goal of this approach is to find the probability that the given task set is schedulable during a mission time L under the specified error model. This probability is dependent on the error characteristics (the minimum inter-arrival time between bursts, T_E, the possible values for the fault duration l_j, and the probability distribution *f(l)* and can be derived from the conditional probabilities that the task set is schedulable under specific sets of values for these parameters.

The analysis begins with finding the maximum number of error bursts, n, that can hit any task in the task set. Considering the interplay between T_E and l_j a set of sensitivity analyses is performed to derive the minimum inter-arrival times between error bursts (T_E) for each possible combination of n fault durations by assuming the worst-case task executions and error overheads.

One should note that the derived minimum inter-arrival times are actually *upper bounds* which may never be reached. This is due to the nature of the inexact worst-case assumptions, such as the WCETs of the tasks, which correspond to upper bounds rather than exact worst-case values.

The fault duration combinations and the corresponding upper bound T_E values are then used to find the *conditional* probabilities of schedulability which are actually lower bounds for the exact probabilities. Finally, the lower bound probability of schedulability is computed as a cumulative sum of these individual conditional lower bound probabilities, i.e. by unconditioning the probability of schedulability with respect to the fault durations. The steps involved in the methodology are illustrated in Figure 1 and briefly described below.

STEP 1. The analysis begins by finding an upper bound for the maximum number of error bursts that can hit any task in the task set while the task set is still schedulable.

STEP 2. In this step, a set of sensitivity analyses is performed for each combination of n fault durations specified in the probability mass function *f(l)* in order to derive the minimum inter-arrival time between bursts (T_E) under which the task set is still schedulable.

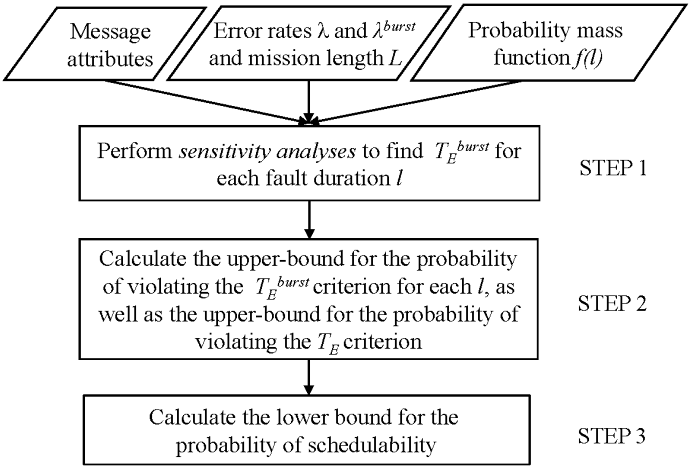
Message attributes
Error rates λ and λ^{burst} and mission length L
Probability mass function $f(l)$
Perform *sensitivity analyses* to find T_E^{burst} for each fault duration l
STEP 1
Calculate the upper-bound for the probability of violating the T_E^{burst} criterion for each l, as well as the upper-bound for the probability of violating the T_E criterion
STEP 2
Calculate the lower bound for the probability of schedulability
STEP 3

Fig. 1. Task scheduling – Methodology overview

STEP 3. The goal of this step is to derive the probabilities that the actual inter-arrival times between bursts will not be shorter than the calculated minimum inter-arrival times by taking into account λ and the mission time *L*.

STEP 4. Finally, based on the probability mass function *f(l)*, as well as the derived probabilities for each fault duration combination, the cumulative probability of schedulability is derived.

Worst-Case Task Response Time Analysis under Error Bursts. In this subsection, a worst-case response time analysis is presented that identifies whether a given task set is schedulable when affected by faults of *random durations* and a minimum inter-arrival time T_E. One should note that if the fault duration is greater than or equal to the minimum inter-arrival time between bursts, every burst can start before the end of the previous one, hence the bursts can potentially affect the whole mission time. If this is the case or if the fault duration is greater than the minimum inter-arrival time of the task whose worst-case response time is to be calculated, the schedulability of this task cannot be guaranteed.

The main differences between the error characteristics in the traditional single error model and the burst model are:

- An error burst may consist of multiple errors
- An error burst may affect multiple tasks

Hence, the worst-case scenario required for calculating the worst-case response times is not the same in case of error bursts as compared to the model introduced in [6].

The set of bursts interfering with a task τ_i, i.e., arriving after the release of τ_i, is denoted by $\{\beta_j | j=1,2,...,n\}$.

Definition 1. *The **worst-case error overhead** E_i^j for a task τ_i caused by an error burst β_j is the largest amount of time required by the task alternates τ_k^{alt}, $\tau_k^{alt} \in \Gamma$, to recover from the effects of burst β_j, in the interval between the release time of task τ_i and its completion.*

Remark 1.1. *An important observation is that, while the worst-case error overhead accounts for all the alternates required for recovery (including the successful ones), it excludes all the primaries, since, although affected by errors, those are already taken into account in the traditional part of the response time analysis* [7,8], *i.e., C_i and* $\sum_{j \in hp(i)} \lceil R_i / T_j \rceil C_j$.

Remark 1.2. *Another observation following Definition 1 is that, in the general case, a burst causes its worst-case error overhead when its interference (i.e., overlap) with the executions of the first and last task it affects is minimized. Hence, it has to start ε before the completion of the first affected task, and end ε after the start of the last affected task.*

The worst-case error overhead for task τ_i caused by an error burst β_j under a burst with length l_j is:

$$E_i^j = \max_{k \in hep(i)}(C_k^{alt} + \sum_{m \in hep(k)} C_m^{alt} + \alpha) \quad \cdot \tag{1}$$

where

$$\alpha = \begin{cases} 0, & if\, k \neq h \text{ and } C_h - (l_j - \varepsilon) \geq C_h^{alt} \\ l_j - \varepsilon + C_h^{alt} - C_h, & if\, k \neq h \text{ and } C_h - (l_j - \varepsilon) < C_h^{alt} \\ l_j - \varepsilon, & otherwise \end{cases}$$

and τ_h is the highest priority task in the task set Γ and ε is an arbitrary small positive real number.

The total interference I_i experienced by a task τ_i is the sum of the maximum interference caused by the higher priority tasks, I_i^{hp}, and the maximum interference caused by error bursts I_i^{err}.

$$\forall \tau_i \in \Gamma, I_i = I_i^{hp} + I_i^{err} . \tag{2}$$

Note that I_i^{hp} is given by the traditional response time analysis [7,8]:

$$I_i^{hp} = \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j . \tag{3}$$

Consequently, the worst-case error interference that needs to be accounted for, in the response time analysis, is obtained by the summing up the worst-case error overheads, E_i^j, of each error burst β_j that is assumed to interfere with task τ_i 's execution. In this case, the maximum interference caused by the error bursts with a minimum inter-arrival time T_E on task $\tau_i \in \Gamma$ in the interval (0, R$_i$] is:

$$I_i^{err} = \sum_{j=1}^{\left\lceil \frac{R_i}{T_E} \right\rceil} E_i^j \,. \tag{4}$$

Hence, the equation that gives the worst-case response time for a task τ_i under error bursts is:

$$R_i = C_i + I_i^{hp} + I_i^{err} \,. \tag{5}$$

Probabilistic Schedulability Bounds. We assume that, during a mission, if the actual shortest interval between any two error bursts *W* is less than the derived minimum inter-arrival time between errors T_E, then the task set is unschedulable. Hence, the probability of schedulability for a T_E value derived for a fault duration combination *Pr(U|combo$_i$)*, is equal to *Pr(W < T_E)*, i.e., the probability of schedulability of a given task set is translated to the derivation of the probability that, during the mission time *L*, no two consecutive error bursts arrive with an inter-arrival time shorter than the derived T_E. Once the probabilities of schedulability (or the upper bound for the probability of unschedulability) for the T_E values derived for each fault duration combination is calculated, the probabilities of the fault duration combinations extracted from the probability mass function *f(l)* are used to calculate the cumulative probability of schedulability.

2.2 PFTA for CAN Scheduling under Error Bursts

This subsection presents a schedulability analysis for real-time message scheduling on CAN, and a sensitivity analysis in order to derive accurate probabilistic schedulability guarantees for fault-tolerant real-time messages. The schedulability analysis presented in this subsection extends the existing CAN response time analysis [9,10,11,12,13] to cope with burst errors modeled with an improved accuracy that enables the specification of a range of new parameters including e.g., fault duration and intensity.

In this section a distributed real-time architecture is assumed that consists of sensors, actuators and processing nodes communicating over CAN. The communication is performed via a set of periodic messages. For the sake of generality, a message M_i is assumed to include N_i frames, hence the worst-case transmission time C_i of the message in an error-free scenario is:

$$C_i = N_i \times f^{\max} \times \tau_{bit} \,. \tag{6}$$

where f^{max} is the maximum frame size in number of bits, and τ_{bit} is the time it takes to transmit a single bit on CAN. However, the analysis presented in this section applies to the particular case of single frame messages as well.

While CAN communication is non-preemptive during the frame transmissions, messages composed of more than one frame can preempt each other at frame boundaries. Additionally, the non-preemptiveness of message frames may cause a higher priority message to be blocked by a lower priority message for at most one frame length, if the high priority message is released during the transmission of a lower priority frame. This priority inversion phenomenon can affect all messages except the lowest priority one, and only once per message period, before the transmission of the first message frame [14].

Each fault may affect the system for certain duration. Depending on the duration of a fault and the minimum inter-arrival time between errors within a fault, a fault can materialize into a burst of errors, only a single error, or no error at all during its length. However it is assumed that at least one error occurs during each fault exposure, since analysis assumes the worst-case scenario. For the sake of presentation, the term *error burst* is used for both error bursts and single errors. The duration of the faults is very much domain specific, and in this paper, it is assumed that the information regarding the probability distribution of the fault durations is available. Errors may occur any time during the fault as long as they satisfy the minimum inter-arrival time condition derived from the sensitivity analyses.

We assume that each error in each message frame is detected as soon as it occurs by the built in CAN error detection mechanisms and upon each error in a frame, an identical frame to the erroneous frame is scheduled for re-transmission following the error frame. Other error model related parameters that are assumed to be given are the rate that the observed system is hit by errors caused by independent faults λ and the mission time L of the system. This section assumes that at most one burst may hit any message instance hence T_E is equal to the largest period of all the messages in the message set.

Methodology Overview. The ultimate goal of this approach is to find the probability that the message set is schedulable under a given fault and error hypothesis. The methodology is outlined in the following steps, and illustrated in Figure 2.

STEP 1. In this step, a series of sensitivity analyses is performed for each l in the probability mass function *f(l)* in order to derive the minimum inter-arrival times of errors within error bursts, T_E^{burst}, for which the message set is guaranteed to be schedulable.

STEP 2. In this step, first an upper bound for the probability of violating the minimum inter-arrival time requirement between errors within a burst, T_E^{burst}, for each fault duration l is calculated. Then, this probability bound on the *fault duration* is unconditioned and an upper bound for the probability of violating the minimum inter-arrival time requirement between errors within bursts under faults of random length, during the whole mission is derived. In this step, separately, an upper bound for the probability of violating the minimum inter-arrival time requirement between error bursts, T_E, during the whole mission is derived.

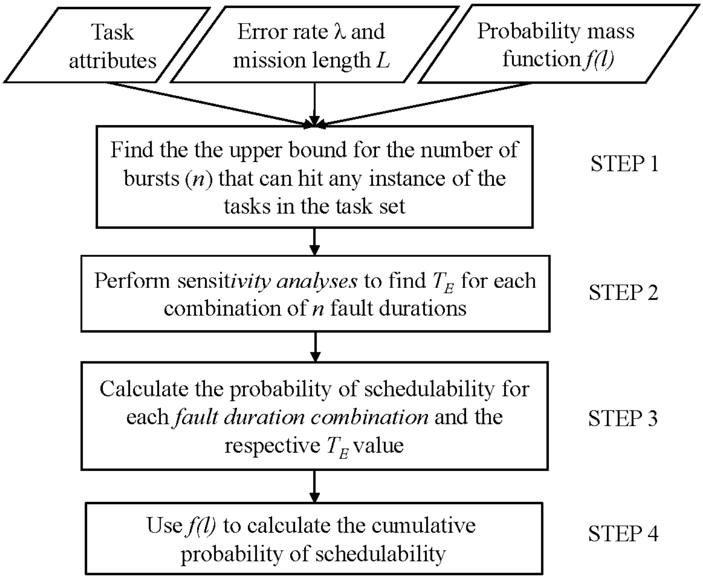
Task attributes
Error rate λ and mission length L
Probability mass function f(l)
Find the the upper bound for the number of bursts (n) that can hit any instance of the tasks in the task set
STEP 1
Perform sensitivity analyses to find T_E for each combination of n fault durations
STEP 2
Calculate the probability of schedulability for each fault duration combination and the respective T_E value
STEP 3
Use f(l) to calculate the cumulative probability of schedulability
STEP 4

Fig. 2. Message scheduling – Methodology overview

STEP 3. Finally, in the last step, an upper-bound for probability of unschedulability, i.e. lower bound for the probability of schedulability, which is shown to be the union of the upper-bounds of the probabilities of at least one occurrence of any two bursts arriving less than T_E apart *or* at least one occurrence of any two errors within a burst, arriving less than T_E^{burst} apart during a mission of length L is calculated.

In the next subsection, a schedulability analysis under error bursts is presented which is the main tool to perform the outlined analysis.

CAN Response Time Analysis under Error Bursts. We present a worst-case response time analysis that identifies whether a given message set is schedulable when affected by error bursts caused by faults with a given duration l and a combination of error inter-arrival time thresholds (minimum inter-arrival time of error bursts T_E and errors within a burst T_E^{burst}). The presented worst-case response time analysis is based on the worst-case response time analysis of CAN under periodic messages and sporadic faults introduced by Tindell et al. [7]. In this work, we use the maximum error interference, I_i^{err} , in the equation used for calculating the worst-case response time of message M_i:

$$R_i = C_i + J_i + I_i^{err} + B_i + \sum_{j \in hp(i)} \left\lceil \frac{q_i + J_j + \tau_{bit}}{T_j} \right\rceil C_j \,. \tag{7}$$

Assuming the burst error model, the worst-case response time calculations will differ in the following cases depending on the minimum inter-arrival time of the errors within an error burst T_E^{burst}:

CASE 1. $T_E^{burst} \leq (e^{max}+f^{max})\tau_{bit}+l$: in this case, if the errors within an error burst occur with a separation of T_E^{burst}, it may not be possible to transmit any frame between any two consecutive errors during the burst. Therefore, the worst-case error overhead I_i^{err} in becomes:

$$I_i^{err} = (f^{\max} + e^{\max})\tau_{bit} + l\,. \tag{8}$$

The error overhead includes the transmission time of the largest frame in the worst-case scenario, i e., when the first error in the burst hits its last bit. The other components of error overhead are the transmission time of the largest error frame and the whole duration of the fault, since in the worst-case, no frame can be transmitted during this time. The largest message frame and the largest error frame in Equation 8 are the frames before and after the error burst respectively.

CASE 2. $T_E^{burst} > (e^{max}+f^{max})\tau_{bit}+l$: *i*n this case, one or more frames can successfully be transmitted between two errors within an error burst. Therefore only certain sections during the exposure to the fault may contribute to the error induced overhead. The worst-case error overhead, I_i^{err}, in this case, is given by:

$$I_i^{err} = (f^{\max} + e^{\max})\tau_{bit} + \left\lfloor \frac{l}{T_E^{burst}} \right\rfloor (e^{\max}\tau_{bit} + (T_E^{burst} - e^{\max}\tau_{bit} - \varepsilon) \bmod f^{\max}\tau_{bit} + \varepsilon) + x \tag{9}$$

where $T_E^{burst} > 0$, $\varepsilon < \tau_{bit}$ and

$$x = \begin{cases} a, & if\, a \leq \left\lfloor \frac{l}{T_E^{burst}} \right\rfloor b \\ \left\lfloor \frac{l}{T_E^{burst}} \right\rfloor b, & if\ a > \left\lfloor \frac{l}{T_E^{burst}} \right\rfloor b \end{cases}$$

a and b are given by:

$$a = l \bmod T_E^{burst}$$

and

$$b = f^{\max}\tau_{bit} - (T_E^{burst} - e^{\max}\tau_{bit} - \varepsilon) \bmod f^{\max}\tau_{bit} + \varepsilon$$

The error overhead in this case includes the transmission time of the largest frame, the largest error frame, and the error overhead *during l*. Note that in this case, the error overhead during l is strictly less than the fault duration l, however, Equation 9 is written in a general form and can be used for both cases. The first term $(f^{max}+e^{max})\tau_{bit}$ in Equation 9 gives the worst-case error overhead caused by the first error in the burst and is equal to the sum of the largest message frame and the largest error frame.

The second term gives the worst-case error overhead caused by a single error during the burst (except the first error) multiplied by the maximum number of errors that can occur during the error burst minus one (the first error) assuming that the errors arrive with an exact inter-arrival time of T_E^{burst} . The product term $\lfloor l / T_E^{burst} \rfloor$ of the second term in Equation 9 gives the maximum number of errors that can occur during an error burst minus one. The product term $e^{max}\tau_{bit} + (T_E^{burst}-e^{max}\tau_{bit} - \varepsilon) \bmod f^{max}\tau_{bit} + \varepsilon$ of the second term includes the transmission time of the largest error frame and the largest message frame the error may hit. The last term x in Equation 9 gives the additional overhead caused by the errors whose relative arrival times are larger than (T_E^{burst}). One should note that, the error overhead for a single error arrived with the minimum inter-arrival time (T_E^{burst}), plus the additional overhead per error caused by late arrivals can at most be equal to $(f^{max}+e^{max})\tau_{bit}$. Therefore, the worst-case value for x is equal to either the total amount of time that can be distributed to the error inter-arrival times for late arrivals (a), or the difference between the overhead assuming all errors hit the largest possible message in the last bit and the overhead assuming all errors arrive with the minimum inter-arrival time between errors within a burst $\lfloor l / T_E^{burst} \rfloor b$, whichever is smaller.

We have assumed that all successfully transmitted frames between two errors in a burst have the maximum frame size. If these frames are shorter than the maximum frame size, the error related interference may be larger than the value calculated by Expression 9. However, this increase in the error interference is bounded by the total sum of the differences between the actual frame sizes and the maximum frame size, i.e, the increase in the error interference is never larger than the cumulative reduction in the frame sizes. Hence, the worst-case response time analysis holds for the general case when message frame sizes are less than maximum, i.e., the analysis never calculates an optimistic value.

Probabilistic Schedulability Bounds. We assume that, during a mission, if the actual shortest interval between any two error bursts W is less than the minimum inter-arrival time between errors T_E, or if the actual shortest interval between any two errors within a burst W^{burst} is less than the minimum inter-arrival time between errors within a burst T_E^{burst} then the message set is un-schedulable. The T_E value is assumed to be equal to the largest period in the message set and the T_E^{burst} value is derived for each fault duration in the probability mass function *f(l)*. Hence, the probability of unschedulability is equal to the union of two probabilities, (i) *Pr(W* < T_E*)* and (ii) the probability of violating the minimum inter-arrival time requirement between errors within a burst caused by bursts of *random* length during the *whole* mission.

2.3 End-to-End Response Time Analysis

In this subsection we present a unified end-to-end response time analysis for fault tolerant distributed real-time systems consisting of tasks executing on nodes and exchanging messages over CAN network, under error bursts. The proposed analysis joins the results for processor scheduling with CAN message scheduling presented above, while taking into account the fault manifestations specific for each scenario as described in the subsections 2.1 and 2.2. Figure 3 illustrates the system model where tasks are executed on two nodes in an event-triggered manner under the FPS scheduling policy, and exchange messages on CAN network. Our goal is to determine the worst-case response time of an end-to-end *transaction* consisting of two tasks on different nodes exchanging one message on CAN.

The derivation of the end-to-end response time for a transaction is illustrated in Figure 4 where task *A* is executing on one node and sends a message *m1* to task *B* on a different node. What needs to be taken into account here is the *jitter inheritance* between nodes and network. In task (or message) scheduling, the *Response Time Jitter* - J_i is the maximum time distance between the response times for any two consecutive task (or message) instances τ_i^k and τ_i^{k+1}, and it is calculated as $J_i = \max\left|R_i^k - R_i^{k+1}\right|$.

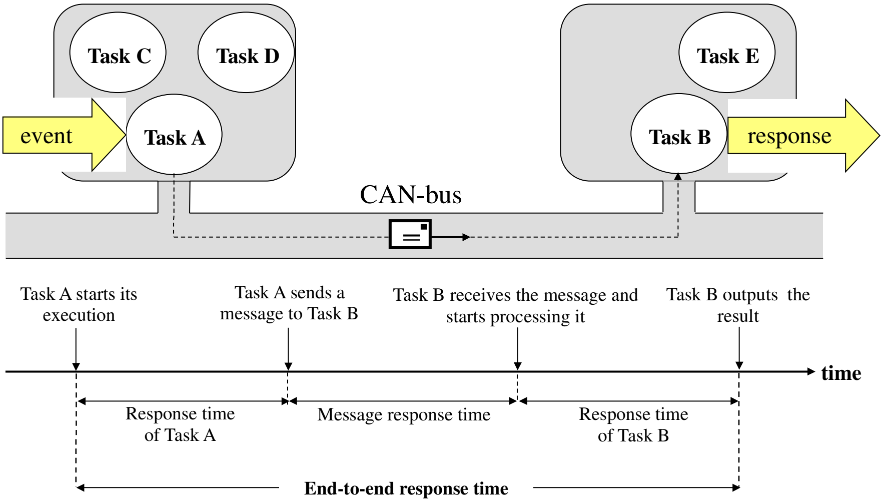

Fig. 3. System overview

Consequently, in the worst-case, the maximum response time jitter experienced by a task τ_i (or a message m_i) is given by:

$$J_i = R_i^{\max} - R_i^{\min} \,. \tag{10}$$

where R_i^{max} is the worst-case response time of τ_i and R_i^{min} is its best case response time given by [15]: $R_i^{\min} = C_i + \sum_{j\in hp(i)} \left\lceil \frac{R_i^{\min}}{T_j} - 1 \right\rceil C_j$.

Hence, the jitter inherited by the message *m1* from the sending task A is $J_{m1} = R_i^{\max} - R_i^{\min}$. Similarly, the jitter inherited by the receiving task *B* from message *m1* is $J_B = R_{m1}^{\max} - R_{m1}^{\min}$. The error interference terms I_A^{err} ,I_B^{err} and I_{m1}^{err} are derived as described in Equations 4 and 9 respectively.

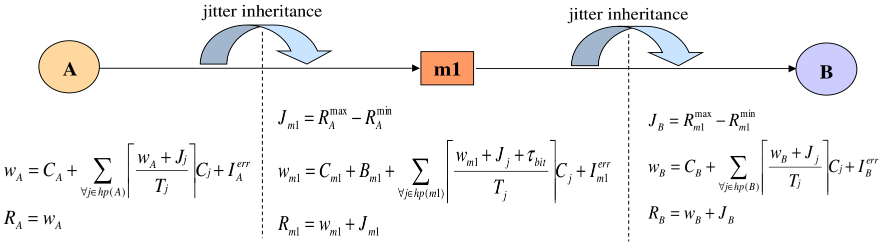
jitter inheritance
jitter inheritance
A
m1
B
$J_{m1} = R_A^{\max} - R_A^{\min}$
$J_B = R_{m1}^{\max} - R_{m1}^{\min}$
$w_A = C_A + \sum_{\forall j\in hp(A)} \left\lceil \frac{w_A + J_j}{T_j} \right\rceil C_j + I_A^{err}$
$R_A = w_A$
$w_{m1} = C_{m1} + B_{m1} + \sum_{\forall j\in hp(m1)} \left\lceil \frac{w_{m1} + J_j + \tau_{bit}}{T_j} \right\rceil C_j + I_{m1}^{err}$
$R_{m1} = w_{m1} + J_{m1}$
$w_B = C_B + \sum_{\forall j\in hp(B)} \left\lceil \frac{w_B + J_j}{T_j} \right\rceil C_j + I_B^{err}$
$R_B = w_B + J_B$

Fig. 4. End-to-end response time for one transaction A->m1->B

End-to-End Probabilistic Schedulability Bound. We assume that the schedulability of a task sets on separate nodes and the schedulability of messages on CAN do not affect each other's schedulability. Hence the end-to-end probability of schedulability of a transaction, $Pr\ (S_{transaction}) = Pr\ (S_{start\text{-}node}) \cup Pr\ (S_{CAN}) \cup Pr\ (S_{end\text{-}node})$, is equal to the multiplication of each individual probability of schedulability: $Pr\ (S_{transaction}) = Pr\ (S_{start\text{-}node}) \quad Pr\ (S_{CAN}) \quad Pr\ (S_{end\text{-}node})$.

5 Conclusions

Design of dependable distributed real-time systems demands advances in both dependability modeling and scheduling theory at node and network level in tandem, to provide system level guarantees that potential error scenarios are addressed in an effective as well as efficient manner. In this work, we have introduced a schedulability analysis framework for dependable networked real-time systems. We presented a sufficient end-to-end analysis that accounts for the worst-case interference caused by error bursts on transactions consisting of tasks scheduled on different nodes under the preemptive FPS policy, and exchanging messages on CAN. The proposed analysis introduces significant improvements over existing works in many aspects, including a more elaborate and realistic error model that relaxes the previous assumptions. Further, we have outlined an overview on how to derive probabilistic scheduling guarantees from the stochastic behaviour of errors by performing a joint schedulability – and sensitivity analysis.

References

1. Stankovic, J.A., Ramamritham, K.: Hard Real-Time Systems Tutorial. IEEE Computer Society Press (1988)
2. Thekkilakattil, A., Dobrin, R., Punnekkat, S.: Preemption Control using CPU Frequency Scaling in Real-Time Systems. In: 18th International Conference on Control Systsems and Computer Science (2011)
3. Navet, N.: Controller Area Networks: CAN's use within Automobiles. IEEE Potentials, 12–14 (1998)
4. Pop, F., Baron, O.-A., Cristea, V.: Rescheduling Services for Reliable Distributed Systems. In: 18th International Conference on Control Systsems and Computer Science (2011)
5. Casparsson, L., Rajnak, A., Tindell, K., Malmberg, P.: Volcano – a Revolution in On-board Communication. Volvo Technology Report. 98-12-10 (1998)
6. Burns, A., Punnekkat, S., Strigini, L., Wright, D.R.: Probabilistic Scheduling Guarantees for Fault-Tolerant Real-Time Systems. In: Dependable Computing for Critical Applications, pp. 361–378 (1999)
7. Audsley, N.C., Burns, A., Richardson, M.F., Tindell, K., Wellings, A.J.: Applying New Scheduling Theory to Static Priority Pre-emptive Scheduling. Software Engineering Journal 8, 284–292 (1993)
8. Joseph, M., Pandya, P.: Finding Response Times in a Real-Time System. The Computer Journal - British Computer Society 29, 390–395 (1986)
9. Tindell, K., Burns, A., Wellings, A.: Calculating Controller Area Network (CAN) Message Response Times. Control Engineering Practice 3, 1163–1169 (1995)
10. Broster, I., Burns, A., Rodriguez-Navas, G.: Probabilistic analysis of CAN with faults. In: 23rd IEEE Real-Time Systems Symposium, pp. 269–278 (2002)
11. Broster, I.: Flexibility in Real-Time Communication. Department of Computer Science, University of York (2003)
12. Navet, N., Song, Y.Q., Simonot, F.: Worst-Case Deadline Failure Probability in Real-Time Applications Distributed over Controller Area Network. Journal of Systems Architecture, 607–617 (2000)
13. Many, F., Doose, D.: Scheduling Analysis under Fault Bursts. In: IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 113–122 (2011)
14. Di Natale, M.: Scheduling the CAN Bus with Earliest Deadline Techniques. In: IEEE Real-Time Systems Symposium, pp. 259–268 (2000)
15. Bril, R.J., Steffens, E.F.M., Verhaegh, W.F.J.: Best-Case Response Times of Real-Time Tasks. In: 2nd Philips Workshop on Scheduling and Resource Management, pp. 19–27 (2001)