Real-Time Guarantees for CAN Traffic

Ulrich Klehmet, Thomas Herpel, Kai-Steffen Hielscher, Reinhard German
Department of Computer Science 7
(Computer Networks and Communication Systems)
University of Erlangen-Nürnberg
Martensstraße 3
D-91058 Erlangen, Germany
Email: {klehmet, herpel, hielscher, german}@informatik.uni-erlangen.de

***Abstract*—Modern cars comprise a multitude of electronic features which are implemented in tens of communicating control units. To connect these in-car embedded systems, the CAN bus offers a sustainable performance, hence it is used as a widespread communication infrastructure, even for safety critical applications. However, CAN media access is priority based and performed competitive and non-preemptive. Thus, assessing the worst case end-to-end delay is inevitable in order to provide safe and efficient operation of functions with hard real-time properties. In this paper, we use the analytical method of Network Calculus to determine guaranteed upper bounds for transmission delays of all CAN priorities. We demonstrate the applicability of our approach by investigating current real-life CAN communication data from the German car manufacturer Audi.**

I. Introduction

The number of electronic control units (ECUs) for various purposes inside new cars has grown constantly in the recent years. Besides entertainment and infotainment, active and passive occupant protection and driver assistance gained remarkably from this development process. Safety critical functions, e.g. ESP (Electronic Stability Program) or ACC (Adaptive Cruise Control) often require data about the current state of the vehicle aggregated from several control devices. Fast, reliable and robust data exchange between the incorporated ECUs is a necessary prerequisite for efficient execution of safety relevant functions and avoidance of dangerous driving situations. CAN (Controller Area Network) is a widespread automotive communication system with a sustainable performance to fulfill these demands. While the FlexRay bus [1] – the major competitor to CAN – implements TDMA (Time Division Multiple Access) and thus exhibits deterministic behavior of data transmission, media access at CAN is priority based and non-preemptive which is why immediate processing of transmission requests can not be guaranteed. Predicting the delay one node has to wait before it can transmit a message of a certain priority is difficult and depends on the number of higher priority data which is sent simultaneously. Mean values for these performance measures, as obtained from stochastic modeling approaches like Queuing Theory, are of minor interest, as they are not sufficient to predict whether hard real-time deadlines are met in any case. The reliable operation and avoidance of system malfunctions with catastrophic effects mainly depend on the worst case performance of the communication infrastructure. Therefore, we use the analytical method of *Network Calculus* to evaluate automotive CAN based networks with respect to timing aspects of data transfer. The application of this deterministic modeling technique yields upper bounds for end-to-end delays of all CAN priorities, which are inevitable to assess reliability and timeliness of hard real-time systems in all possible scenarios of operation.

The paper is organized as follows: Section II presents previous approaches to analyze timing aspects of CAN and in section III the main features of CAN are explained briefly. Section IV introduces the basic idea of Network Calculus. Using this theory, we construct the modeling elements and determine delay bounds. Section V applies these findings to real-life CAN communication data from Audi. Finally, section VI summarizes our investigations.

II. Related Work

Various studies investigate timing aspects of CAN. [2] uses the *Earliest Deadline First* algorithm for a global optimized schedule of traffic, whereas in [3] *Response Time Analysis* is applied. [4] and [5] extend the previous findings, [6] models CAN as timed automata and [7] presents a tool-supported evaluation. These approaches require an a-priori bus-wide traffic schedule to analyze the real-time properties. Due to asynchronous wake-up of nodes along a CAN bus, such a global schedule can never be anticipated. To overcome this, the evaluation procedure that we propose requires just the statically assigned CAN identifiers and the transmission cycles of each message as input. We do not need to know the global traffic schedule to determine upper delay bounds for all priorities.

III. The CAN Bus

We give only a brief introduction to CAN, a detailed description can be found in [8], [9].

CAN is a bus system aiming at automotive applications, offering data rates of up to 1 Mbps. A serial line architecture is used with the logical bus levels 0 (dominant) and 1 (recessive, also “bus idle state”) and complementary bit-stuffing after five successive bits with equal level. CAN communication does not employ explicit sender/receiver addressing, each station listens to the bus for messages of interest. The message content

978-1-4244-1645-5/08/$25.00 ©2008 IEEE

is described by the unique identifier (ID) in the header field of a CAN frame (cf. figure 1). The IDs have to be assigned statically during the design phase of the bus system to avoid ambiguity in the interpretation of the frame content. For all following investigations, we assume 130 bit CAN frames, including a maximum number of 19 stuff bits.

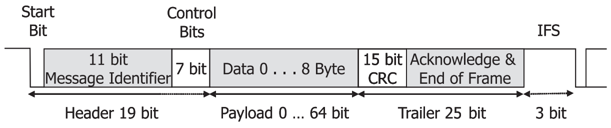

Figure 1. CAN Frame Structure

CAN implements *CSMA/BA* (Carrier Sense Multiple Access with Bitwise Arbitration), a node starts transmission only if the bus has been recessive for at least 6 bit times (CS phase). Due to bit-stuffing, a potential sender receives at least one dominant bit during the CS phase. Each station listens to the bus while sending data, if a collision occurs where one station tries to send a recessive bit but receives a dominant bit, it will notice that another station is sending simultaneously and will immediately stop its own transmission, which makes the arbitration non-destructive. Transmission of a CAN frame starts with one dominant start bit immediately followed by the ID from the most to the least significant bit. Thus, the identifiers create an implicit hierarchy of priorities. If more senders start to send simultaneously, the sender transmitting the frame with the highest ID has to send a recessive 1 bit that is overwritten by a dominant 0 bit first due to the binary encoding of the message IDs in the frame header. It will detect the collision and stop transmission. This process continues until only the node transmitting the frame with the lowest ID and thus with the highest priority remains sending.

IV. Network Calculus

Network Calculus is a theory for deterministic performance evaluation, focusing on worst case behavior of traffic flows. The aim is to determine lower and upper bounds for performance measures, e.g. end-to-end delays. By means of these analytic bounding values it is possible to limit bursts or to schedule traffic appropriately. In this section, we introduce only the basic features of Network Calculus. A comprehensive overview can be found in [10]–[12].

A. Basic Theoretical Foundations

The main modeling elements of Network Calculus are the *arrival curve* and the *service curve*.

Let F be a flow (bits, messages, etc.) into a system S and let $x(t)$ be the amount of data of F arriving in the time interval $[0, t]$ and $y(t)$ the amount of data leaving S in the time interval $[0, t]$. By definition, $x(0)$ is zero, and $x(t) \geq x(s)$ for all $t \geq s$.

The *arrival curve* $\alpha(t)$ defines an upper bound for the cumulative input flow $x(t)$.

Definition 1 (Arrival Curve)**.** *Let $\alpha(t)$ be a non-negative, non-decreasing function. A flow F is constrained by or has the arrival curve $\alpha(t)$ iff $x(t) - x(s) \leq \alpha(t-s)$ for all $t \geq s \geq 0$.*

Example. *A commonly used arrival curve is the token bucket constraint: $\alpha_{r,b}(t) = b + rt$ for $t > 0$ and zero otherwise. Figure 2 shows the shape of $\alpha_{r,b}(t)$. b denotes an instantaneous data burst and r the average arrival rate.*

The service curve $\beta(t)$ aims at determining service guarantees, i.e. the minimum output $y(t)$ of the system S.

Definition 2 (Service Curve)**.** *A system S offers a (minimum) service curve $\beta(t)$ to the cumulative input function $x(t)$, which is a non-negative, non-decreasing function with $\beta(0) = 0$. Processing $x(t)$ according to $\beta(t)$ yields a guaranteed lower bound for the cumulative output function $y(t)$:*

$$y(t) \geq \inf_{s \leq t} \{x(s) + \beta(t-s)\}.$$

Example. *One commonly used service curve is the rate-latency function $\beta(t) = \beta_{R,T}(t) = R \cdot max\{(t-T), 0\}$. It reflects a service element offering a minimum service of rate R after a worst case latency of T. Figure 2 shows a rate-latency service curve $\beta_{R,T}(t)$.*

Constructing the appropriate arrival curve $\alpha(t)$ and service curve $\beta(t)$ allows to determine the maximum delay d of data in S.

Theorem 1 (Delay Bound)**.** *Assume a flow constrained by the arrival curve $\alpha(t)$ passing a system with the service curve $\beta(t)$. The maximum delay d is given as the supremum of all possible delays of data, i.e. is defined as the supremum of the horizontal deviation between the arrival and service curve:*

$$d \leq \sup_{s \geq 0} \{\inf\{\tau : \alpha(s) \leq \beta(s+\tau)\}\}.$$

Example. *For a system with input according to a token bucket and rate-latency output, the delay bound $d \leq T + b/R$ is determined as depicted in figure 2.*

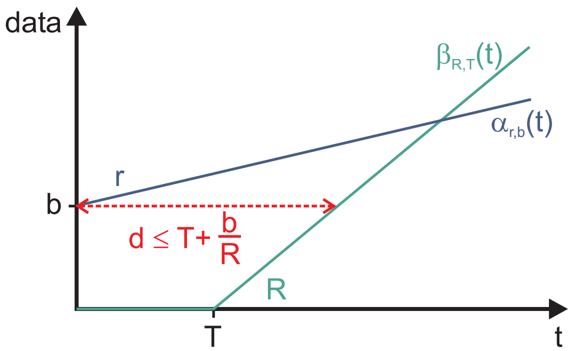

Figure 2. Example for Arrival Curve, Service Curve and Delay Bound

B. CAN Input Data

For investigations of real-life CAN traffic, the German car manufacturer Audi provided us with drivetrain CAN data of several up-to-date model series (cf. section V). As shown later, knowing the ID and the cycle times of the periodically sent

CAN messages is sufficient to apply Network Calculus for determination of worst case delays. In the remaining part of the paper the distinction between "ID" and "priority class" (or simply "priority") is made. Ordering the messages in the data set from lowest to highest ID in use, we abstract from the actual identifier and introduce a ranking of priorities where class 0 corresponds to the lowest ID and hence has highest priority, class 1 complies with the next-lowest ID, class 2 with the third-lowest ID and so on.

C. Generation of Arrival Curves

A central issue of performance evaluation with Network Calculus is the generation of appropriate arrival curves for data traffic. Recall that CAN media access is performed cyclic and successful transmission depends on the priority of the message to be sent compared to the priority of data which attempts transmission simultaneously.

Let N denote the number of the lowest priority class and $n \in \{0, 1, \dots, N\}$ be a priority of one particular CAN frame. Basically, there are three types of priorities: The "own" priority n (a certain message belongs to), the "higher" priorities 0 to $n-1$ (which win the competitive media access) and the "lower" priorities $n+1$ to N (which this message dominates at media access, hence we can neglect them in the following). Since the arrivals of higher priority data affect the transmission of lower priorities, we need to consider them explicitly by forming an individual arrival curve (cf. section IV-D). It represents the cumulative arrivals of messages during a specified time interval at which no media access for class n is possible due to higher prioritized traffic. The highest priority class 0 has to be treated separately, since here no arrival curve for higher priorities exists.

The generation of arrival curves for the actual drivetrain CAN is based on the worst case assumption that at each discrete point of time, all messages with matching cycle period occur at once. Hence, the lower the priority of the message, the more higher prioritized data may access the bus at the same time, prolongating the time until media access for the particular message is successful. This approach is valid, since performance evaluation with Network Calculus aims at determination of upper delay bounds in a worst case scenario.

The data provided by Audi shows that a message of priority n is sent cyclic in cycles of length $T_{c_n} \in \mathbb{N}_0$. A new message of priority n will be sent immediately after times $k \cdot T_{c_n}, k \in \mathbb{N}_0$, i.e. at integer multiples of T_{c_n}. Without loss of generality, we can construct a "Master Cycle" as the least common multiple of all T_{c_n}. Beyond this time span, the arrivals of messages of all priorities will repeat in the same sequence, thus it is sufficient to investigate one single Master Cycle in order to include the whole CAN data traffic. The arrival curve for a message of class n can be written as the following *step function*:

$$\alpha_n(t) = \left\lceil \frac{t}{T_{c_n}} \right\rceil \cdot l,$$

where $l = 136$ bits denotes the maximum frame length including the 6 bit CS time.

The cumulative number of arrivals of messages with priority higher than n can be calculated as

$$\hat{\alpha}_n(t) = \sum_{i=0}^{n-1} \alpha_i(t) = \sum_{i=0}^{n-1} \left\lceil \frac{t}{T_{c_i}} \right\rceil \cdot l.$$

In addition, we determine a linear function $\bar{\alpha}_n(t)$ as an upper bound for the cumulative higher priority arrivals $\hat{\alpha}_n(t)$ as

$$\bar{\alpha}_n(t) \quad = \quad n \cdot l + \sum_{i=0}^{n-1} \frac{l}{T_{c_i}} \cdot t.$$

Defining $b_n := n \cdot l$ and $r_n := \sum_{i=0}^{n-1}(l/T_{c_i})$ allows us to write this function in a token bucket form:

$$\bar{\alpha}_n(t) = b_n + r_n \cdot t.$$

D. Determination of Service Curves and Calculation of Delay Bounds

Now we determine actual service curves for a CAN bus with 500 kbps data rate and compute upper bounds for delays at media access. Due to non-preemption, a transmission in progress is not interrupted, thus a frame of arbitrary priority has to wait until data currently on the bus is completely sent. As we assume 130 bit CAN frames and a CS phase of 6 bit times, the worst case sending time of such non-preempted data is $T = l/R = 136\,\text{bits}/500\,\text{kbps} = 0.272\,\text{ms}$.

Thus, for the rate-latency service curve $\beta_{R,T}(t)$ it holds:

$$\begin{aligned} \beta_{R,T}(t) &= R \cdot \max\{(t - T), 0\} \\ &= 500\,\text{kbps} \cdot \max\{(t - 0.000272\,\text{s}), 0\}. \end{aligned}$$

We compute the guaranteed worst case delays for traffic of different priority. Therefore, we use the arrival and service curve only and propose the following approach:

Let $x_j(t)$ denote the input data and $y_j(t)$ the output data at time t per controller for priority $j \in \{0, \cdots, N\}$. The procedure starts with input $x_0(t)$ and output $y_0(t)$. Theorem 1 tells us that delay

$$d_0 \le \sup_{t \ge 0}\{\inf\{\tau : \alpha_0(t) \le \beta(t + \tau)\}\}$$

with $\alpha_0(t)$ = step-function-type arrival curve of input $x_0(t)$ and $\beta(t) = \beta_{R,T}(t) = 500\,\text{kbps} \cdot \max\{(t - 0.000272\,\text{s}), 0\}$. Then we have to determine the maximum delay d_1 for the next lower priority frames of the same controller station. However, it would not be correct to take the same service curve $\beta(t)$, because the frames of priority 1 will be served only after sending of frames of priority 0 is finished. We follow the approach of aggregate traffic modeling as presented in [13], [14] and construct the service curve

$$\beta_1(t) := \max\{(\beta(t) - \bar{\alpha}_1(t)), 0\}.$$

We use $\bar{\alpha}_1(t)$ instead of $\hat{\alpha}_1(t)$ to guarantee that $\beta_1(t)$ is a non-decreasing function. The maximum delay d_1 of frames of priority 1 is

$$d_1 \leq \sup_{t\geq 0} \{\inf \{\tau : \alpha_1(t) \leq \beta(t+\tau) - \bar{\alpha}_1(t+\tau)\}\}.$$

Going on in this way from priority j to the next lower one $j+1$ we must build a new service curve by diminishing the previous one by the cumulative arrival curve $\bar{\alpha}_{j+1}(t)$ for frames with priority $0, 1, \ldots, j$. Thus, at each priority step only two sorts of frames are considered: the frames of present priority j and the sum of all higher priority frames – all frames of lower priority have no impact besides maybe a single frame just in service which has already been modeled by the "non-preemptive" scheduling in the latency component T of $\beta_{R,T}(t)$. The following procedure summarizes our approach: $\forall j \in \{0, 1, \ldots, N\}$ and $t \in [0, \infty]$:

$$\begin{aligned}
\alpha_j(t) &= \left\lceil \frac{t}{T_{C_j}} \right\rceil \cdot l \\
\bar{\alpha}_j(t) &= b_j + r_j \cdot t \\
\beta_j(t) &= \max\{(\beta(t) - \bar{\alpha}_j(t)), 0\} \\
&= \max\left\{ \left((R - r_j) \cdot \left(t - \frac{RT + b_j}{R - r_j} \right) \right), 0 \right\}.
\end{aligned}$$

Again, we get a rate latency service curve with rate $R'_j := R - r_j$ and latency $T'_j := (RT + b_j)/(R - r_j)$. Lastly, we get

$$d_j \leq \sup_{t\geq 0} \{\inf \{\tau \geq 0 : \alpha_j(t) \leq \beta_j(t+\tau)\}\}.$$

Due to the actual shape of the arrival curve $\alpha_j(t)$ and the service curve $\beta_j(t)$ in our case, this inequation can be solved geometrically by calculating the intersection point of $\beta_j(t)$ with the height of the first step of the arrival curve $\alpha_j(t)$, which is exactly $l = 136$ bits. This yields the following results:

$$\begin{aligned}
d_j &\leq \frac{l}{R'_j} + T'_j \\
&= \frac{l + RT + b_j}{R - r_j} \\
&= \frac{(j+2) \cdot l}{R - \sum_{i=0}^{j-1} (l/T_{c_i})}.
\end{aligned}$$

The application of this equation is sufficient to calculate the maximum delay d_j for each priority j, a given data rate R and message length l. The only input data needed for each priority class is the cycle length T_{c_j}.

V. Application to Audi CAN Traffic

Audi supported our investigations with real-life CAN data sets for various current series. We applied our approach to data for three different models, in the following referred to as model A, B and C. The cycle times for the model dependent varying number of active CAN priority classes are depicted in figure 5.

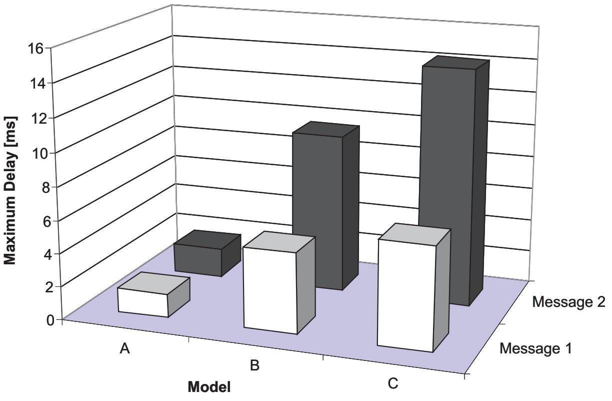

Figure 3. Comparison of Worst Case Delays in Model A, B and C

The computed worst case delays over the message priority are shown in figure 4 for model A, B and C. We can contrast these results to maximum communication delays for FlexRay. In FlexRay, the worst case is that the dedicated time slice for media access has just elapsed when an ECU is ready to send data. The controller then has to wait for the next TDMA slot before it can access the bus again. Thus, the actual cycle time is an upper delay bound for *all* FlexRay messages. Considering a typical TDMA cycle, e.g. 10 milliseconds, our method allows to determine which priority classes in a given CAN schedule will experience a guaranteed delay *below* the maximum FlexRay TDMA delay. The results from figure 4 reveal that the first 21 (model A), 23 (model B) and 22 (model C) CAN priorities are kept beneath this threshold and hence show a better real-time transmission behavior in a worst case operation scenario.
In figures 5 and 4, two priority classes are highlighted in white and black. They represent messages which are semantically identical, i.e. contain the same information in each model. The priority class they are assigned to and the number of higher priority classes differ from one model to the other. Hence, we expect different worst case delays for the same content in varying traffic load and scheduling scenarios. Figure 3 depicts the worst case delays for the two messages we selected for reasons of comparison. The values deviate significantly between the three models, emphasizing the strong influence of the actual priority class and the traffic load of higher prioritized data on the maximum delay at CAN media access. The numerical values are listed in table I.

Table I
CAN Parameters and Worst Case Delays in Model A, B and C

	Message 1			Message 2		
Model	ID	Priority	Delay [ms]	ID	Priority	Delay [ms]
A	130	4	1.446	131	5	1.787
B	1088	14	4.836	1344	24	9.635
C	1088	17	6.140	1344	30	14.269

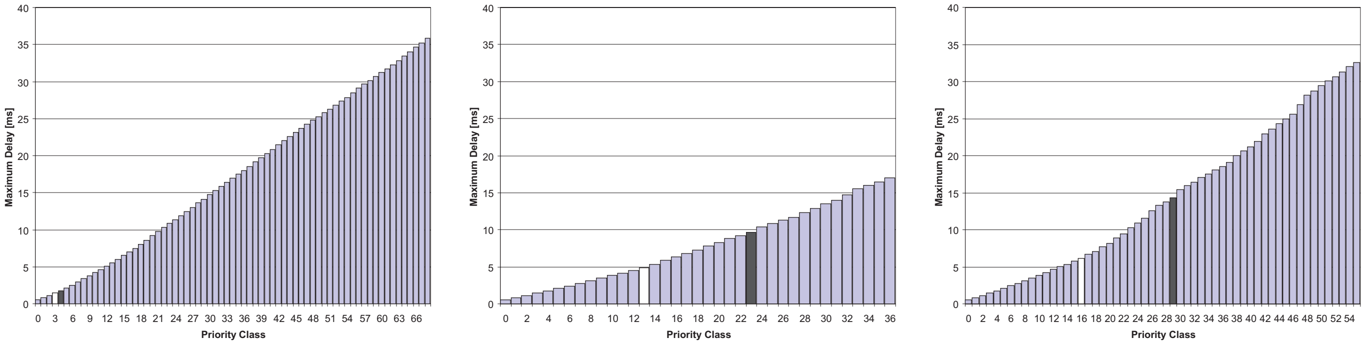

Figure 4. Worst Case Delays Ordered by Message Priority - Model A, B and C

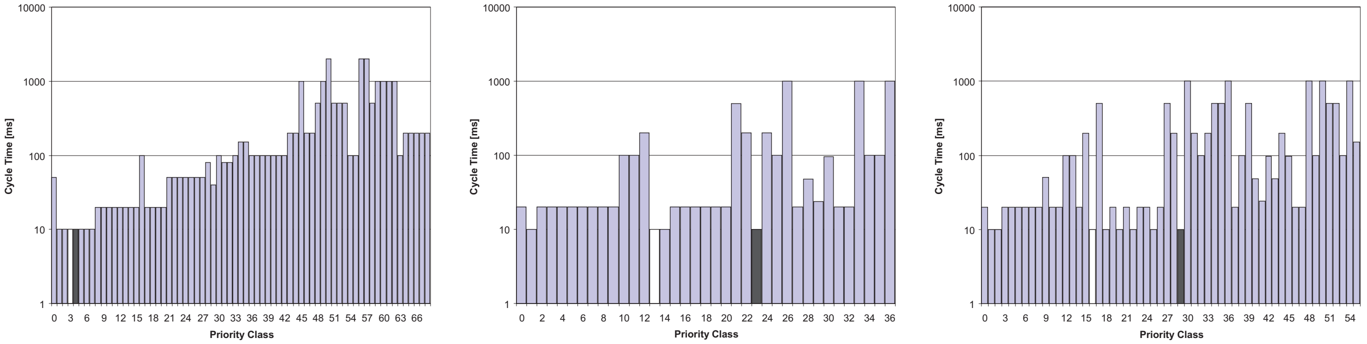

Figure 5. Cycle Times of Drivetrain CAN Traffic - Model A, B and C

VI. CONCLUSIONS

Applying Network Calculus yields reasonable results for performance evaluation of state-of-the-art automotive communication systems. The delay bounds for CAN traffic support the decision whether hard real-time transmission properties of safety critical functions can be fulfilled using a given CAN infrastructure and allow for comparing the worst case transmission behavior of CAN traffic with schedules for alternative FlexRay bus system solutions. The approach is straightforward and simply applicable. It requires just the CAN IDs and the cycle times of periodically sent CAN messages as input parameters and provides an analytical method to determine delay bounds for all CAN priorities with a closed and intuitive formula. As the amount of electronics increases drastically in modern cars, application of formal methods becomes inevitable for successful system development. Engineers from Audi appreciate our investigations, they can rely on sound estimations for communication constraints when developing and integrating new functions. Thus, the development department at Audi will apply the method for future vehicle projects to ensure both reliable CAN operation and avoidance of excessive CSMA/BA delays for safety critical data transfer.

REFERENCES

[1] FlexRay Consortium, *FlexRay Communications System Protocol Specification*, version 2.1 ed., 2005.

[2] P. Richardson, L. Sieh, A. Elkateeb, and P. Haniak, “Real-time Controller Area Networks (CAN) – managing transient surges,” *Integr. Comput.-Aided Eng.*, vol. 9, no. 2, pp. 149–165, 2002.

[3] K. Tindell and A. Burns, “Guaranteed Message Latencies for Distributed Safety Critical Hard Real-Time Networks,” Dept. Computer Science, Univeristy of York, Tech. Rep. YCS 229, 1994.

[4] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien, “Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised,” *Real-Time Syst.*, vol. 35, no. 3, pp. 239–272, 2007.

[5] A. Davare, M. DiNatale, and Q. Zhu, “Period Optimization for Hard Real-time Distributed Automotive Systems,” in *Proceedings of the 44th IEEE/ACM Design Automation Conference*, June 2007.

[6] J. Krakora and Z. Hanzalek, “Verifying Real-Time Properties of CAN bus by Timed Automata,” in *In FISITA 2004 - World Automotive Congress, Barcelona*, May 2004.

[7] A. Hamann, R. Racu, and R. Ernst, “Formal Methods for Automotive Platform Analysis and Optimization,” in *In Proc. Future Trends in Automotive Electronics and Tool Integration Workshop (DATE Conference), Munich*, March 2006.

[8] International Standard ISO 11898, *Road vehicles – Interchange of Digital Information – Controller Area Network (CAN) for High Speed Communication*, 1st ed. International Organization for Standardization, 1994, ISO Reference Number ISO 11898:1993(E).

[9] W. Lawrenz, Ed., *CAN Controller Area Network*, 4th ed. Heidelberg: Hüthig Verlag, 2000.

[10] J.-Y. Le Boudec and P. Thiran, *Network Calculus*. Springer Verlag LNCS 2050, 2001.

[11] R. L. Cruz, “A calculus for network delay, Part I: Network elements in isolation,” *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 114–131, 1991.

[12] ——, “A calculus for network delay, Part II: Network analysis,” *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 132–141, 1991.

[13] M. Fidler and V. Sander, “A parameter based admission control for differentiated services networks,” *Computer Networks*, vol. 44, pp. 463–479, 2004.

[14] E. Wandeler and L. Thiele, “Real-time interfaces for interface-based design of real-time systems with fixed priority scheduling,” in *EMSOFT ’05: Proceedings of the 5th ACM international conference on Embedded software*. New York, NY, USA: ACM Press, 2005, pp. 80–89.