In-vehicle communication networks : a literature survey

IN-VEHICLE COMMUNICATION NETWORKS:

A LITERATURE SURVEY

Uğur Keskin
Technische Universiteit Eindhoven (TU/e)
Den Dolech 2, 5600 AZ Eindhoven, The Netherlands
ukeskin@tue.nl

JULY 28, 2009

ABSTRACT

The increasing use of electronic systems in automobiles instead of mechanical and hydraulic parts brings about advantages by decreasing their weight and cost and providing more safety and comfort. There are many electronic systems in modern automobiles like antilock braking system (ABS) and electronic brakeforce distribution (EBD), electronic stability program (ESP) and adaptive cruise control (ACC). Such systems assist the driver by providing better control, more comfort and safety. In addition, future x-by-wire applications aim to replace existing braking, steering and driving systems. The developments in automotive electronics reveal the need for dependable, efficient, high-speed and low cost in-vehicle communication. This report presents the summary of a literature survey on in-vehicle communication networks. Different in-vehicle system domains and their requirements are described and main in-vehicle communication networks that have been used in automobiles or are likely to be used in the near future are discussed and compared with key references.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

4WD	4 Wheel Drive
ABS	Antilock Braking System
ACC	Adaptive Cruise Control
ACK	Acknowledgment
ASC	Automatic Stability Control
ASIC	Application Specific Integrated Circuit
ASR	Anti-Slip Regulation
AUTOSAR	Automotive Open System Architecture
CAN	Controller Area Network
CC	Cruise Control
CD	Compact Disc
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSMA	Carrier Sense Multiple Access
CSMA/CA	CSMA/Collision Avoidance
CSMA/CD	CSMA/Collision Detection
CSMA/CR	CSMA/Collision Resolution
D2B	Digital Data Bus
DLC	Data Length Code
DM	Deadline Monotonic
DVD	Digital Versatile Disc
EBD	Electronic Brakeforce Distribution
ECU	Electronic Control Unit
EDC	Electronic Damper Control
EDF	Earliest Deadline First
EEPROM	Electrically Erasable Programmable Read-Only Memory

EMI	Electromagnetic Interference
EOF	End of Frame
EPS	Electronic Power Steering
ESP	Electronic Stability Program
FTT-CAN	Flexible Time-Triggered Controller Area Network
GPS	Global Positioning System
I/O	Input/Output
ISO	International Standards Organization
LIN	Local Interconnect Network
MEDL	Message Descriptor List
MOST	Media-Oriented System Transport
NGU	Never-Give-Up
NTU	Network Time Unit
OFP	Optimized Frame Packing Algorithm
OSI	Open Systems Interconnection
RAM	Random Access Memory
RM	Rate Monotonic
ROM	Read-Only Memory
RTR	Remote Transmission Request
SA	Simulated Annealing
SAE	Society of Automotive Engineers
SM	System Matrix
SOF	Start of Frame
SP	Straightforward Solution
TDMA	Time Division Multiple Access
TTCAN	Time-Triggered Controller Area Network
TTP	Time-Triggered Protocol
TTP	Timed Token Protocol
VAN	Vehicle Area Network

1. INTRODUCTION

The safety, comfort and performance requirements and thus the functions of in-vehicle systems have been increasing steadily. As a result, there has been an increase in the number of electronic control units (ECU) and communication signals with more complex interrelations between them to meet the requirements. This result reveals the need for more robust, dependable and efficient high-speed in-vehicle communication. Such systems contain hard real-time messages that have strict timing requirements. The exchange of these messages in the network is conducted by the in-vehicle communication protocols that can be classified as event-triggered, time-triggered and hybrid networks. These networks are expected to schedule real-time messages to provide timeliness in communication for a healthy run of the system.

Today automotive electronic systems contain electronic control units, sensors and actuators that are distributed and embedded in vehicles. The use of such systems is increasing as mechanical and hydraulic parts are replaced or new functions are added to them. Most of these are real-time systems that possess strict timing requirements in terms of deadlines and response time jitter. For instance, in modern cars nearly 2500 signals are exchanged by up to 70 electronic control units [1][2], both of which tend to increase with higher demands on safety, comfort, functions and cost. Electronic control units, referred to as the main processing units, of automotive systems form several networks that have different properties, regarding their architectures, services and functions depending on communication requirements. One of the most important requirement is the providing these networks with the integrity and interoperability.

This report gives a literature survey of automotive domains and in-vehicle communication networks by reviewing past and recent studies on examining and comparing communication protocols, developing new approaches and improving real-

time performance. Developments in the automotive area are discussed, and demands on new in-vehicle networks are revealed. The report will also mention studies on these networks designed for future embedded automotive electronic systems.

The outline of the report is organized in the following manner: In Section 2, performance attributes for automotive systems are defined and main automotive domains and their requirements based on these attributes are explained. Section 3 describes time and event-triggered approaches and compares them with giving related references. In Section 4, the properties of main wired in-vehicle networks are summarized with comparing each other. Finally, Section 5 gives some final remarks about in-vehicle communication.

2. AUTOMOTIVE DOMAINS

The introduction of electronic systems into automobiles, owing to the production of small electronic devices during 1960s, gave rise to the rapid development in automotive applications. Not only automotive electronic systems but also the size of software embedded in these systems have made considerable advances during the last two decades with bringing about the increase in memory size and performance as presented in [3][4]. In addition, these developments have provided the use of smaller automotive systems with less mechanical and hydraulic back-up that result in less weight and lower cost as well as performance benefits. This phenomenon is explained in [3] and [4] by illustrating the evolution of automotive electronics resulting in better performance in engine control and safety as well as lower cost and smaller size for system implementations.

As stated earlier the increasing number of embedded automotive electronic systems and their functions reflects the increasing complexity for in-vehicle networks. At the same

time, the demands on performance, reliability, cost and time-to-market are getting tighter. Thus, designing such systems is becoming more important and difficult, demanding new design mechanisms for both hardware and software architectures of automotive systems.

This section deals with performance attributes that are widely used in literature to define the real-time communication in vehicles. Moreover, main automotive domains and their requirements will be explained with regard to defined performance attributes.

2.1. Performance Attributes

Efficient, dependable and high-speed (especially for systems that require high data rates) communication in automotive systems is crucial to provide better real-time performance in terms of timeliness, bandwidth utilization and communication delay. To satisfy these diverse demands, different in-vehicle communication protocols are currently in use (e.g. Controller Area Network (CAN), Local Interconnect Network (LIN), Byteflight, Media Oriented System Transport (MOST)) or upcoming for future automobiles (e.g. Time-Triggered Controller Area Network (TT-CAN), Time Triggered Protocol (TTP) and FlexRay). To relate both communication requirements of in-vehicle systems and characteristics of communication protocols, particular performance attributes have been defined in the literature. In general these can be classified as *flexibility*, *predictability*, *dependability*, *composability*, *extensibility* and *network bandwidth*. These terms are so general that they should be defined more precisely to use properly in the context of performance interpretation of in-vehicle communication networks.

In [6] flexibility is defined as “the ability to make decisions at runtime”. In addition to that, in [7] flexibility is explained based on several important attributes, such as *design flexibility*, *configuration and reconfiguration flexibility*, *network traffic flexibility*,

integration flexibility, *test flexibility*, *functional flexibility* and *just-on-time flexibility*. Among these attributes, network traffic flexibility can be explained as the ability of the communication architecture to adapt to network traffic changes. Similar to network traffic flexibility, just-on-time flexibility is defined as the ability of the communication architecture to support any change quickly to meet strict message deadlines (timeliness). In a similar way, in this report flexibility is considered as the ability to adapt to changing network conditions (network load and configuration, sporadic traffic and interrupts) in terms of timeliness (satisfying message deadlines), response time (communication delay) and bandwidth utilization [5][6][7]. Response time[1] is defined as the time elapsed between the arrival of a message for transmission and the completion of the transmission, which is the successful read of the message by a receiver node. Moreover, bandwidth utilization relates the percentage of the use of the bandwidth with message transmissions. In this context, these are considered as measures treating the flexibility as a performance attribute. Timeliness, however, is an important real-time requirement of in-vehicle networks that must be satisfied.

Predictability is another performance attribute that can be specified as the capability to predict temporal behavior of the communication performed in a network. Predictability can be expressed in terms of response times or the exact times at which the messages will be sent and received. There is a trade-off between flexibility and predictability. Performing the real-time communication based on a static time schedule (e.g. TTP) makes the system more predictable in temporal domain. However, it decreases flexibility by making the communication static and not capable of adapting to changing or unexpected traffic conditions in the network.

Dependability is defined in [6] and [8] as the ability to provide service with verified reliability and is expressed as one of four characteristics (functionality, performance,

[1] In this definition response time is exclusively considered as the latency of a message transmitted between ECUs within a single network.

cost and dependability) of a computing system. Moreover, dependability is shown as a tree with three main elements: (*i*) attributes (availability, reliability, safety, confidentiality, integrity and maintainability), (*ii*) means (fault prevention, fault tolerance, fault removal and fault forecasting) and (*iii*) threats (faults, errors and failures). The attributes of dependability related in [8] can be described as followings:

- Availability: The readiness of the service for usage,
- Reliability: The continuity of the service,
- Safety: The ability to avoid harmful consequences,
- Confidentiality: The ability to prevent unauthorized access to information (it may also referred to as security of information),
- Integrity: The consistency of system states and their transitions,
- Maintainability: The ability to be repaired updated or modified.

In the context of this report, dependability is considered as the result reliability (fault tolerance, error detection and recovery), safety and availability. The other attributes integrity and maintainability will not be considered in this report. In addition, confidentiality is considered for only wireless communication.

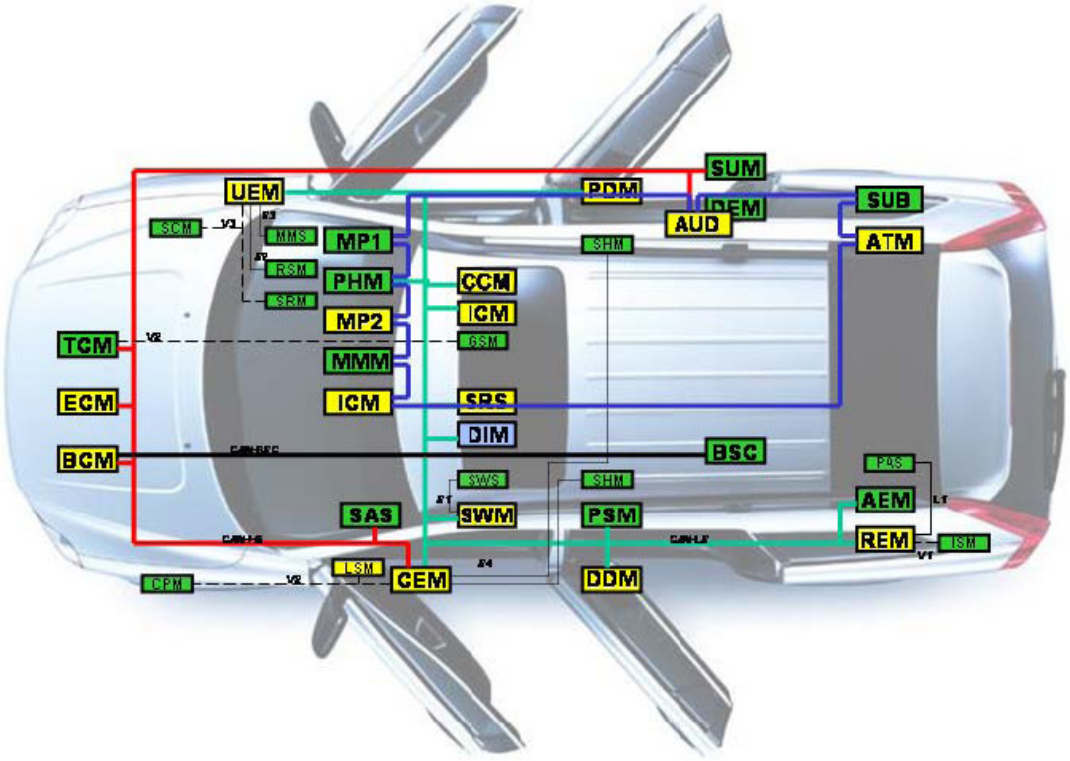
Fourthly, composability is defined as the ability to integrate systems, while validating subsystems' timeliness and testability properties [9]. In this context, the attribute of composability is considered and discussed as temporal composability. It relates to whether system integration would result in any modification and change of temporal properties (i.e. arrival and response time information) of the messages exchanged within the network. More precisely, a lower degree of dependency between the change of system configuration and temporal properties means higher temporal composability.

The attribute extensibility refers to the ability to allow easy network extension from the communication point of view. In this context, network extension is used to relate adding new node components and introducing new messages to the network.

And finally, network bandwidth (data rate) of a network is the available speed that the transmission medium serves. Different in-vehicle protocols propose different bandwidths that play an important role on deciding to employ which in-vehicle networks for a particular application.

2.2. Automotive Domains and Requirements

In-vehicle embedded systems can be divided into five main functional domains [1][10] based on corresponding properties such as, architectures, services and constraints: *powertrain*, *chassis*, *body* and *telematics/wireless* and emerging domain *passive safety*. Figure 2.1 [10] gives the Volvo XC90 network architecture, illustrating four main automotive domains where powertrain and chassis nodes are interconnected with CAN, nodes of body domain with LIN and infotainment nodes (a sub-domain of telematics) with MOST networks. The letter M in the figure stands for the term, Module.



Block	Powertrain and chassis
TCM	Transmission control M
ECM	Engine control M
BCM	Brake control M
BSC	Body sensor cluster
SAS	Steering angle sensor
SUM	Suspension M
DEM	Differential electronic M
Block	**Infotainment**
AUD	Audio M
MP1	Media player 1
MP2	Media player 2
PHM	Phone module
MMM	Multimedia M
SUB	Subwoofer
ATM	Antenna tuner M

Block	Infotainment (cont')
ICM	Infotainment control M
Block	**Body electronics**
DDM	Driver door M
REM	Rear electronic M
PDM	Passenger door M
CCM	Climate control M
ICM	Infotainment control M
UEM	Upper electronic M
DIM	Driver information M
AEM	Auxiliary electronic M
SRS	Supplementary restraint system
PSM	Passenger seat M
SWM	Steering wheel M
CEM	Central electronic M

Figure 2.1 Volvo XC90 network architecture [10]

2.2.1. The Powertrain Domain

The powertrain domain mainly includes the processes: generation of power in the engine (engine control) and transmission of it through the gear box to the driving axis and wheels (transmission and gear control). Powertrain domain possesses several complex control mechanisms including high computing complexity. The real-time subsystems forming this domain have frequent data exchanges between chassis and body domain with strict timing requirements. Thus, powertrain systems require a high network bandwidth, high dependability and predictability in communication. Since the systems and the network conditions of this domain are stable and well defined, a low degree of flexibility would be enough to cope with different message traffics and network loads.

2.2.2. The Chassis Domain

The chassis domain has functions of active safety, driving dynamics and assistance which include systems such as ABS (antilock braking system), ESP (electronic stability program), ASC (automatic stability control), ACC (adaptive cruise control), ASR (anti-slip regulation), EPS (electronic power steering), 4WD (4 wheel drive), EDC (electronic damper control) and active suspensions. Similar to powertrain domain, chassis systems have closed loop and advanced real-time control systems that have safety critical applications with strict timing requirements. X-by-wire applications [11] can also be included in this domain because of the similar requirements and services they provide. In [12] the generic term x-by-wire is defined to show the replacement of mechanical and hydraulic automotive systems with electronic counterparts. Automotive terms such as brake, steer, shift, drive or throttle can be substituted for the letter “x”. Figure 2.2 [13][14] shows an example of a steer-by-wire system prototype without any mechanical backup. It consists of steering control units, actuators (steering-wheel actuator and steering actuator) and some sensors to provide angle and torque values as a

feedback to the system and driver. These components are connected via TTP/C bus as an in-vehicle network.

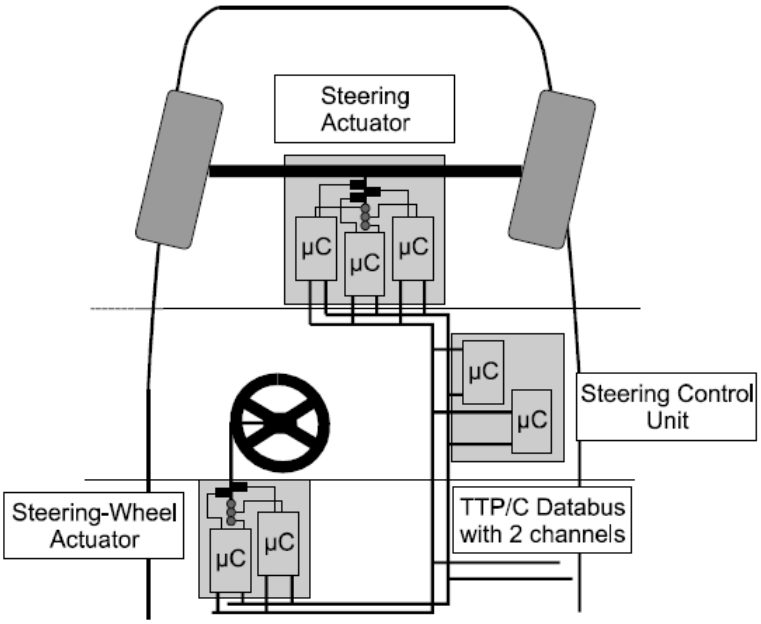


Figure 2.2 A steer-by-wire system prototype [13][14]

Similar to other chassis systems these applications are safety critical. In summary, chassis domain, including the future x-by-wire applications, requires high dependability, high bandwidth and flexibility to some extent. Especially dependability and bandwidth requirements make time-triggered and hybrid approaches likely solutions for in-vehicle networks in this domain.

2.2.3. The Body Domain

The body domain that contains the largest number of ECUs mainly implements body/comfort functions. Air conditioning and climate control, dashboard, wipers, lights, doors, seats, windows, mirrors, locks, cruise control (CC), park distance control

are the main elements that form the body domain. Body domain applications are not safety critical and not all nodes require high bandwidth where the communication mainly depends on sporadic driver/passengers' interaction. The communication in this domain is generally implemented by low cost networks.

2.2.4. The Telematics Domain

The telematics domain consists of multimedia, infotainment and wireless sub-domains. GPS and in-vehicle navigation systems, CD/DVD players, rear seat entertainment, audio systems, monitors and displays are the functions of multimedia and infotainment domain. Moreover, services such as hands-free phones, connection with laptop computers and GPS units and car access systems rely on wireless communication. Moreover, wireless technology in vehicles presents additional functions and services like navigation and traffic information systems, advanced driver assistance, fleet management systems, safety and security systems, diagnostics and maintenance services, voice recognition and wireless internet connection. It is typical for this domain that a huge amount of data is exchanged between systems both in the vehicle and also with the external world. Unlike the embedded real-time networks employed in the previously explained automotive domains, QoS, security and a higher degree of composability and extensibility requirements are more important for the networks in the telematics domain. In addition, because of the need for the transmission of huge and diverse data, high network bandwidth and flexibility are other critical performance requirements.

2.2.5. The Passive Safety Domain

Finally, the passive safety domain employs the systems [17] such as impact and rollover sensors, airbags and belt pretensioners. As serving safety related functions, the networks in safety domain requires high dependability and predictability in addition to high-speed data transmission.

2.3. Classification of In-vehicle Networks

It is apparent from the previous discussion that because of diverse properties and functions, in-vehicle domains have different communication requirements. In 1994, a classification of in-vehicle networks was published by the Society of Automotive Engineers (SAE). According to this, networks are classified exclusively based on bandwidth (data rate) and functions of networks [1][18][19]. In this classification, *Class A* denotes low speed/low cost networks with data rates of less than 10 kb/s and they are mostly dedicated to the body domain. Local Interconnect Network (LIN) [20] and Time-Triggered Light Weight Protocol (TTP/A) are examples of such networks. *Class B* networks, operating at data rates of between 10 and 125 kb/s, are used for general information exchange (i.e. vehicle speed, instrument cluster) and some body domain networks that require higher speed. J1850 [21] and low speed Controller Area Network (CAN-B) are the main examples of this class. Different from above, *Class C* (i.e. high speed CAN (CAN-C) [22]) and *Class D* networks require high speed communication. The data rates of *Class C* networks range from 125 kb/s to 1Mb/s and are used for a wide range of applications, especially in powertrain and chassis (excepting x-by-wire applications) domains. By contrast, data rates in *Class D* networks are up to or higher than 1 Mb/s, and they are mainly used for telematics (for multimedia and infotainment data) and x-by-wire applications. Media-Oriented System Transport (MOST) [23], Digital Data Bus (D2B) [24] and Bluetooth as wireless communication [25] are prime examples of Class D networks for telematics data transmission. In addition to previously mentioned, there are networks that can provide high data rates (more than 1 Mb/s) like Time Triggered Protocol (TTP/C) [26], FlexRay [27][28] and byteflight [25][29][30] protocols that are mainly applied to in-vehicle safety (active and passive) and x-by-wire applications. The figure below relates the comparison of some of the pre-stated communication networks used in automobiles with respect to network bandwidth and communication cost. They are placed in the chart based on their allowable data rates with respect to relative communication cost per ECU. In general, wiring,

microcontrollers and other hardware implementations as well as data overhead and resource consumption determine the cost value [20][23].

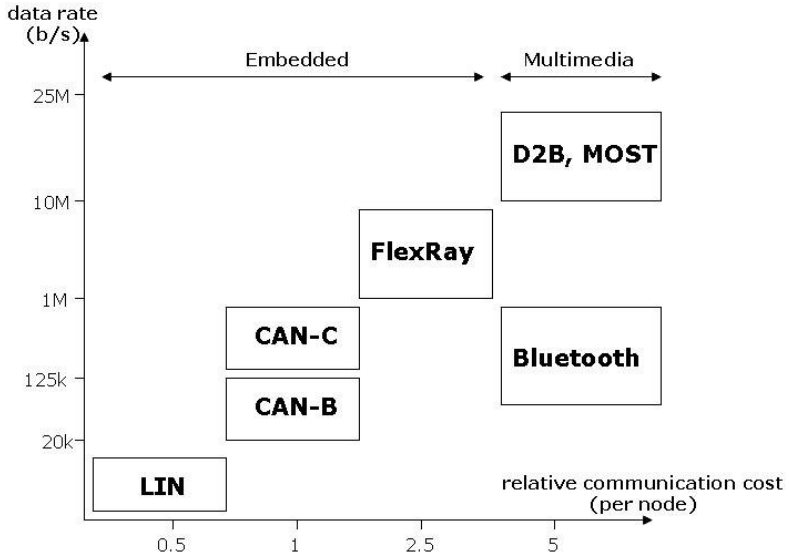


Figure 2.3 Comparison of several in-vehicle network protocols with respect to data rate and communication cost [20]

2.4. In-Vehicle Network Examples

In [10] in-vehicle networks of these functional domains are shown by different network architectures and protocols (i.e. Volvo XC90, BMW 7 series and VW Passat). Figures 2.4 [15] and Figure 2.5 [16] illustrate the network infrastructures of the BMW 7 series and VW Passat. As shown in Figure 2.4, different types of CAN networks (with different data rates), K-CAN, F-CAN, PT-CAN and LoCAN, are used for systems respectively in chassis, powertrain, body and comfort domains. However for multimedia/infotainment systems and passive safety systems MOST and byteflight (SI-BUS) networks are preferred. The interconnection between different networks is provided with a gateway. Similar interconnection is also maintained in VW Passat network infrastructure. As can be seen in Figure 2.5, only CAN and LIN networks are

used, but CAN with different data rates. CAN Antrieb is used for powertrain and chassis systems, whereas CAN Komfort and CAN Infotainment are used for body and multimedia/infotainment systems respectively. In addition to CAN, LIN is also used for body and comfort functions.

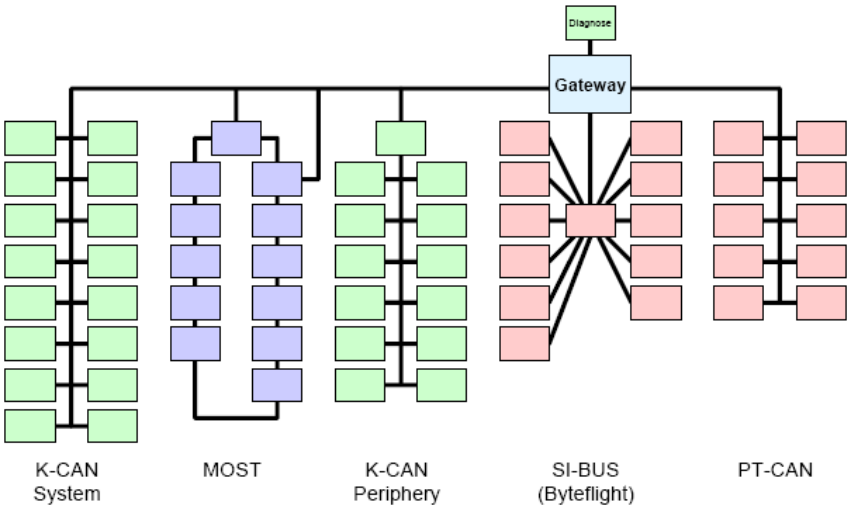


Figure 2.4 BMW 7 Series network infrastructure [10][15]

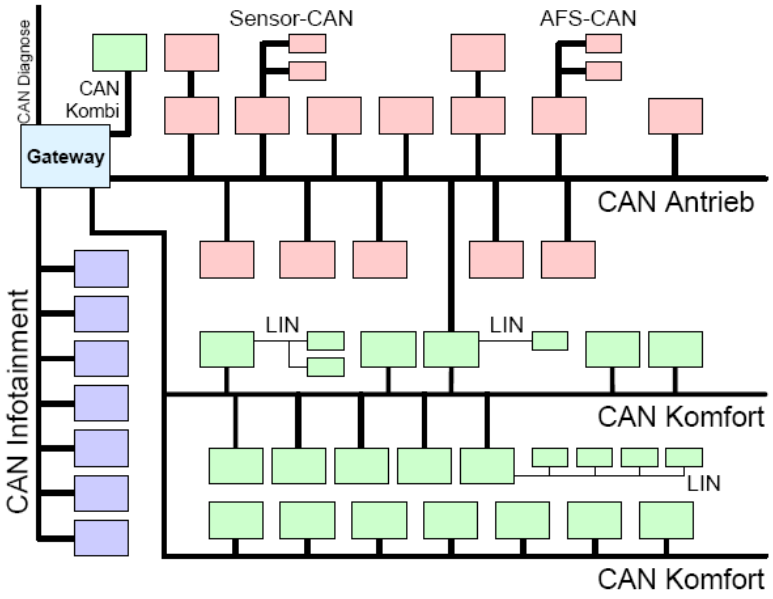


Figure 2.5 VW Passat network infrastructure [10][16]

Additionally, in [10], networking technology details of the BMW 7 series for each functional domain are given in categories such as program size, number of ECUs, messages and cycle time, required bandwidth, safety requirements and bus topology, that are given by Table 2.1. And finally, Table 2.2 summarizes the in-vehicle domain requirements.

Table 2.1 BMW 7 series domain properties in numbers [10]

	Powertrain	**Chassis (Active safety)**	**Body**	**Telematics**	**Passive safety**
Program size	2 MB	4.5 MB	2.5 MB	100 MB	1.5 MB
Number of ECUs	3-6	6-10	14-30	4-12	11-12
Bandwidth	500 Kb/s	500 Kb/s	100 Kb/s	22 Mb/s	10 Mb/s
Number of messages	36	180	300	660	20
Cycle time	10 ms-10 s	10 ms-10 s	50 ms- 2 s	20 ms-5 s	50 ms
Safety requirements	high	high	low	low	very high
Bus topology	bus	bus	bus	ring	star

Table 2.2 In-vehicle domains' communication requirements' matrix

	Flexibility	Predictability	Dependability	Bandwidth	Confidentiality
Powertrain	low	high	high	high	N/A
Chassis	some	high	high	high	N/A
Body/Comfort	some	some	some	low	N/A
Telematics	high	some	low	high	high
Passive Safety	low	high	high	high	N/A

In Table 2.2, chassis domain includes both active safety and x-by-wire systems. Since the attributes of composability and extensibility are common requirements for networks, they are not mentioned in the table. Although they are common requirements for embedded networks, especially these attributes are highly important for the systems in telematics domain. In addition, confidentiality is considered for only wireless communication such as communication between different vehicles (inter-vehicle communication) or between the vehicle and outside world and as given in Table 2.2.

3. TIME AND EVENT-TRIGGERED COMMUNICATION

Bus network protocols can be evaluated under different communication classifications such as time-triggered versus event-triggered [5][31][2][32][33][34] that is the most commonly contrasted one in the literature. In this section, first a brief general model of in-vehicle networks will be given. Secondly, event and time-triggered communication paradigms will be discussed and compared based on defined performance attributes. In addition, the hybrid approach that is the combination of both paradigms will be presented.

3.1 A General Network Model

Several network topologies (e.g. mesh, star, bus, ring and topologies with gateways) can be proposed to provide communication between networked nodes. Currently, because of being simple and versatile (easy system extension and evolution) and having low cost (installation cost and weight saving with less wiring) serial communication with bus networks appears to be an appropriate solution [35]. Figure 3.1 illustrates a fieldbus network architecture, which is an example of a distributed computer control system comprising a bus and nodes (ECUs), each of which consists of a central processing unit (CPU) as host processor, memory (RAM, ROM and EEPROM etc.), I/O interface and communication interface (communication adapter). Moreover, the communication interface compromises a communication controller and a transceiver. Also, nodes can be added to obtain application-specific integrated circuits (ASIC) for acceleration purpose [36].

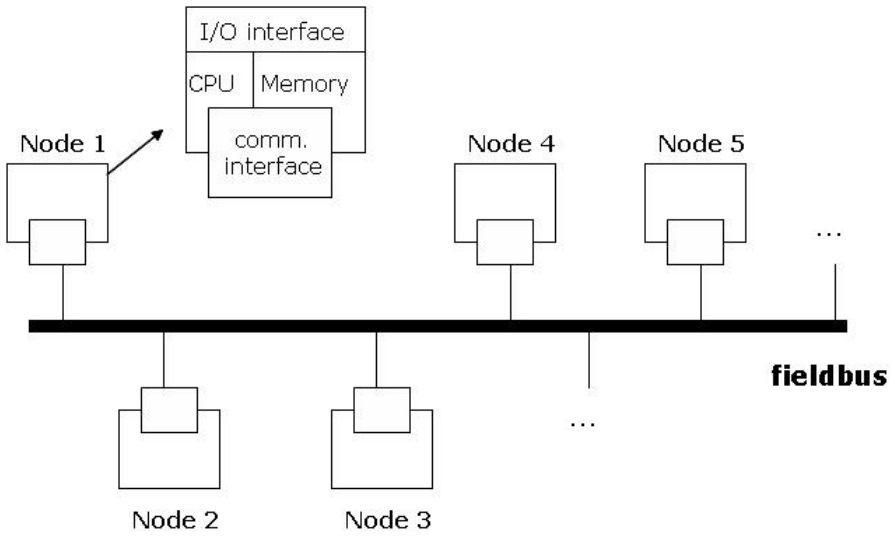


Figure 3.1 Fieldbus network architecture

The example of a shared bus network architecture given above in Figure 3.1 represents a network model of an in-vehicle distributed control system. Such networks are designed and implemented to perform a (set of) specific application(s), examples of which were given in the discussion of automotive functional domains in subsection 2.2. In such domains, there are possibly more than one network which does not need to be homogeneous, i.e. consisting of identical hardware components (CPU, memory) and same task and communication scheduling mechanisms and protocols. And some of these networks may be needed to work together to perform a task in an application having interoperability and communication requirements.

The applications contain a set of real-time software programs (application software components), each of which consists of a set of tasks. A task is a sequence of instructions and it starts after being triggered or getting necessary inputs. Tasks can be pre-emptible or non-pre-emptible. Those tasks, that have strict timing requirements due to the hard real-time nature, are embedded in the processing units in the ECUs. Tasks are scheduled and executed on nodes based on scheduling and resource management mechanisms because of limited processing time and memory. Functions performed by different organizational sequences (called process graphs in [36]) of the tasks, belonging to an application, form the application. Figure 3.2 [36] illustrates the application model explained above and it gives three task graphs comprising an application that involve tasks and messages. In the figure, the arrow relates the message exchange between tasks that are mapped on different nodes. The dashed arrow represents the communication between tasks in the same node. The terms P_i and m_j are used for tasks and messages respectively.

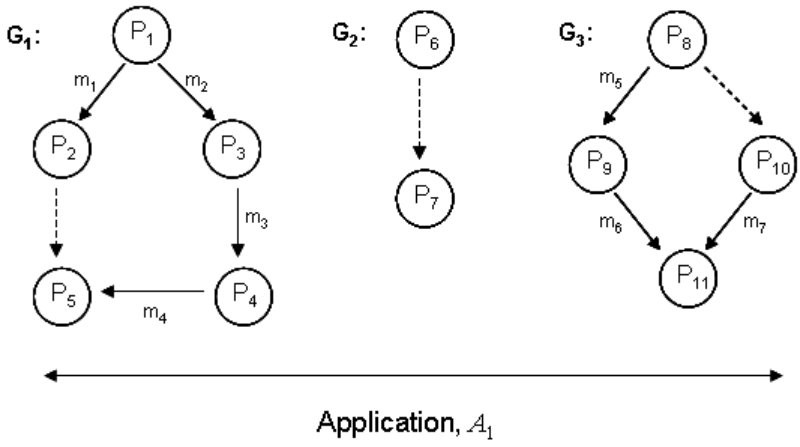


Figure 3.2 Application model involving tasks and messages [36]

As stated earlier, since the software components, or functions, of an application can be available distributed over one or more networks, a communication infrastructure is necessary to provide interaction between different nodes not only within the same network but also within different networks. Nodes generate several real-time signals, periodic or aperiodic, such as control, state, feedback and alarm signals etc. based on task executions or sensor/actuator outputs, and they are transmitted across the network(s), that may be necessary for the execution of tasks in other nodes, while also satisfying timing requirements. In modern cars, such systems may produce an excessive number of signals, and they are generally packed into message frames in order to gain communication medium bandwidth by transmitting less message frame overhead. After the release of the message frame, it is passed to the communication controller to be queued as being ready for transmission. It is transmitted through the transceiver based on the communication scheduling mechanism of the in-vehicle network.

As seen from the above discussion, task and communication scheduling are highly related to each other, which is important for the timeliness behavior of the system. Since the design of such systems is complex, analysis and optimization techniques are

important and necessary. Tasks have to be scheduled on nodes and the communication has to be scheduled on the network so that time analysis of applications involves the schedulability of both tasks and messages. In [36][37] the holistic approaches for time and schedulability analysis are presented. In [37] the authors extend the existing static priority pre-emptive scheduling for distributed hard real-time systems with a TDMA protocol for communication applied to a shared broadcast bus. In [36], multi-network (called multi-cluster in [36]) architecture and an application model are given as a heterogeneous system containing both event and time-triggered clusters that are connected to each other via a gateway. In addition, the multi-cluster optimization problem is defined as having two domains: (*i*) partitioning (assigning a task to the event or time-triggered cluster) and mapping of tasks to the nodes, (*ii*) frame packing. Moreover, in [36] the authors propose a multi-cluster scheduling algorithm, and schedulability analysis method for the event-triggered network. Finally, an optimized frame packing algorithm (OFP), in addition to the other two: straightforward solution (SP) and simulated annealing (SA), is presented, and as an experiment they are applied on an automotive application, vehicle cruise controller, with the aim of comparing them based on execution time and schedulability degree of the algorithms. Thus, they showed that the optimized frame packing algorithm utilized by the proposed schedulability analysis performs better in execution time compared to SA and produce better schedulable solutions compared to SP.

3.2 Communication Paradigms: Event & Time Triggered

The communication scheduling mechanisms can be based on different paradigms such as event-triggered and time-triggered. These paradigms define the basic behaviors of communication protocols. In event-triggered communication, messages are transmitted based on significant events and asynchronous (event-triggered) message transmissions are performed as quickly as possible. Most event-triggered protocols are based on the CSMA/CR (carrier sense multiple access/collision resolution) media access method.

The transmission of messages is performed by bus arbitration based on message priorities to prevent collisions. Because of this property, such bus networks are also called priority buses [1]. Flexibility, extensibility and the ability of quick response to asynchronous events are important advantages of the event-triggered approach. Quick response and message transmission upon occurrences of events give this paradigm a higher degree of flexibility in terms of response time and bandwidth efficiency. Furthermore, because of communication scheduling mechanisms they use (CSMA/CR for CAN, based on message identifiers), this type of networks are easier to be extended. Redesigning a communication schedule for the new configuration is not necessary. Vehicle Area Network (VAN) [38], J1850 and CAN [22][39][40] are the main examples of this paradigm. VAN and J1850 protocols generally used to be employed in the body domain but have recently been replaced by CAN that has been a *de-facto* standard in vehicular communication [1].

In the time-triggered approach, communication between nodes is performed by the progress of time. In other words, message transmission is driven at predefined time instants based on the time division multiple access (TDMA) bandwidth allocation scheme. Since time intervals for message transmissions (access of nodes to bus) are predefined and deterministic, missing messages in the networked system or an error/fault in a node can easily be detected and removed that makes the approach predictable (bounded response times) and dependable. Also, depending on a static schedule served with progress of time results in no need for priority scheduling and arbitration mechanisms as well as bus monitoring (as in CAN), which gives them higher bus bandwidth rates. Yet, this property also makes system change somewhat difficult that adding new nodes and messages to the system requires changing the predefined communication schedule. Furthermore, clock synchronization with high precision is necessary for time-triggered networks. A high degree of predictability which is the result of a predefined and static communication schedule makes fault tolerance mechanisms easier to be implemented for time-triggered networks, which

brings about a higher degree of dependability. Thus, a time-triggered protocol can be proposed as a likely solution especially for real-time systems (i.e. for active safety systems and x-by-wire applications) that require high bandwidth and dependability. For instance, the TTP/C protocol implements the TDMA scheme in which each node access rights to the bus in sequential, predefined and static time instants during consecutive TDMA rounds. Consecutive TDMA rounds form the cluster cycle that repeats itself in a loop during the system run.

To sum up the comparison between event- and time-triggered paradigms, dependency of the communication on a predefined and fixed schedule of a time-triggered network makes it more predictable compared to an event-triggered network. For composability, a change in system configuration in an event-triggered network can result in change of temporal properties of messages. However, in a time-triggered network, message transmission contents is specified and stored into the communication controllers of the nodes during design time (communication schedule construction). This makes the communication temporal properties not dependent on the application software. Thus, a change in system configuration does not affect temporal properties as much as an event-triggered network making time-triggered networks have a higher degree of composability. Moreover, communication schedules impose temporal isolation between the message transmissions of different nodes and temporal isolation together with predictability make the fault tolerance techniques easier to be implemented within the protocol. Furthermore, since there is no need for bus access contention and arbitration, and so bus monitoring in the time-triggered networks, they propose higher network bandwidth rates compared to event-triggered ones. However, this deterministic and predictable behavior makes time-triggered networks less flexible. And, since a change in a network configuration (adding nodes or messages to the network) may result in the need for the redesign of the communication schedule and all its entries, time-triggered networks propose a lower degree of extensibility. Owing to no use of static

communication schedule and the use arbitration mechanisms instead event-triggered networks have a higher degree of flexibility and extensibility.

To respond to diverse requirements of automotive systems, hybrid networks are proposed as a combination of event and time-triggered approaches. In this way, it is aimed to provide some flexibility by also sheltering event-triggered traffic in addition to high dependability, predictability and temporal composability owing to time-triggered communication schedule. However, the static and predefined schedule for time-triggered communication makes hybrid protocols flexible and extensible to some extent. In the hybrid approaches a temporal isolation is needed between these two different traffics. Time-Triggered Controller Area Network (TT-CAN) [22][41][42][43][44][45], FlexRay and byteflight are examples of this approach. In addition, there are some academic protocols, which can be classified as hybrid, such as Flexible Time-triggered Controller Area Network (FTT-CAN) [5][46][47] and Server-CAN [10].

In the presentation [48], the communication protocols CAN and TTP/C as representatives of event and time-triggered approaches are compared with respect to defined criteria by giving also simple scenarios and related results. Section 4 involves the further discussion on the CAN and the TTP/C networks. Table 3.1 summarizes the positive and negative aspects of event and time-triggered communication.

Table 3.1 Comparison of in-vehicle communication paradigms

	Flexibility	**Predictability**	**Dependability**	**Composability**	**Extensibility**
Event-triggered	high	low	medium	low	high
Time-triggered	low	high	high	high	low
Hybrid	medium	high	high	high	medium

It should be noted that the evaluation given in the table above is intended as a general idea only considering the nature of event-triggered, time-triggered and hybrid paradigms. Thus, with additional mechanisms brought in during the design stage an in-vehicle network protocol may show satisfactory performance in one of the attributes that are expected to be weak due to the paradigm it is developed on. For instance, additional error detection and recovery mechanisms make CAN dependable to some extent. Considering its flexibility, the performance of bandwidth utilization and response times depends on network load. For low and average load conditions event-triggered networks perform better but for higher traffic loads performance difference between time and event-triggered networks decreases. Even under high loads, low priority messages suffer from long waiting times to be transmitted in event-triggered networks, whereas in time-triggered networks they are transmitted during reserved window based communication schedules.

4. IN-VEHICLE COMMUNICATION PROTOCOLS

Embedded automotive networks are designed considering a harsh in-vehicle environment mainly caused by noise and electromagnetic interference (EMI). In-vehicle protocols are implemented to provide reliable and available communication under harsh conditions and disturbances. In general, these protocols define both physical and data link layer in the ISO/OSI reference model and they are developed based on some alternative medium access control mechanisms [10]:

- CSMA/CD (carrier sense multiple access / collision detection), e.g. Ethernet,
- CSMA/CR (CSMA / collision resolution), e.g. CAN,
- CSMA/CA (CSMA / collision avoidance),
- TDMA (time division multiple access), e.g. TTP/C,
- FTDMA (flexible TDMA), e.g. Byteflight and FlexRay,

- Distributed solution relying on tokens, e.g. TTP (Timed Token Protocol),
- Centralized solutions by the usage of masters, e.g. LIN and TTP/A.

It is possible that an in-vehicle protocol can use more than one of these alternative mechanisms. For instance, the FTT-CAN protocol uses centralized solutions by employing a master node to schedule time-triggered messages and it additionally reserves bandwidth for event-triggered messages, which implies that FTT-CAN uses FTDMA at the same time. Moreover, during the respective reserved intervals, event-triggered message frames are sent based on the CAN arbitration that uses the CSMA/CR mechanisms.

Based on the network model described at the beginning of Section 3, an embedded real-time in-vehicle network can be defined with three layers of ISO/OSI reference model: physical layer, data link layer and application layer. It is possible to have an additional layer between data link layer and application layer, called middleware layer (communication layer), with the aim of facilitating the integration of different software-based components [1]. The functions of this layer will be explained later in this section.

In Section 4, main wired in-vehicle communication protocols will be summarized considering different properties: bus topologies, bandwidth, communication scheduling and fault tolerance mechanisms etc. Finally, the section summarizes middleware layer for in-vehicle networks.

4.1. In-Vehicle Networks

In this subsection, main wired embedded in-vehicle communication networks will be described. In addition, a multimedia/infotainment protocol that is commonly used in vehicles will be explained briefly. And finally, all described in-vehicle networks will be summarized in a table considering the general network properties in addition to

communication scheduling and fault tolerance mechanisms, synchronization and some additional services they provide.

4.1.1. Local Interconnect Network (LIN)

LIN is a low cost and low speed (20 kb/s) serial bus in-vehicle communication network that is typically used for body/comfort functions. LIN is a time-triggered network and it uses master/slave mechanism, in which the master node manages the message transmissions according to a schedule table by broadcasting a header (frame identifier serving as transmission request) on the bus, and then the slave that possesses the message with this header sends the data. The data field of a LIN frame contains up to 8 B of data. In a LIN network bandwidth reservation is provided by the polling list mechanism of the master node. LIN offers services of bandwidth saving (no response of a slave node to the request of the master node if there is no update for the related data to be sent, so another node can use the bandwidth) and energy saving (sleep modes for nodes). Today, LIN is widely used in the body domain of automobiles because of being simple and low-cost. Yet, in some body domain networks that require higher speed, low-speed CAN (CAN-B) with a bit rate up to 125 kb/s is preferred.

4.1.2. Controller Area Network (CAN)

CAN is a serial, broadcast bus that was developed by Robert Bosch GmbH in the mid-1980s. Subsequently, it became an ISO standard in 1994 [1], and currently it is the de-facto standard for in-vehicle data transmission. CAN is the most widely used automotive communication network with the advantages of providing flexible and robust communication with bounded delay and having low cost and simplicity. It offers different bandwidth rates of up to 1 Mb/s, allowing a maximum of 40-m of bus length at this data rate. As given in Figure 4.1, a standard CAN 2.0A data frame consists of seven fields: start of frame (SOF) bit, 18 bits header, 0-8 byte data, 15 bits cyclic redundancy check (CRC) field, 3 bits acknowledgement slot (ACK), 7 bits end of frame

field (EOF) and last 3 bits intermission frame space. Moreover, header of a frame can be divided in to 3 minor fields that are 11 bits identifier field (29 bits for CAN 2.0B, extended format), remote transmission request (RTR) bit and 4 bits data length code (DLC). A CAN frame can contain up to 8 B of data. The identifier part (11 bit or 29 bit for extended CAN frame format) defines message priorities during arbitration for the bus access. CAN arbitration is based on CSMA/CR mechanism to prevent frame collisions during transmission on the bus. At this point, identifier field, belonging to header of a CAN frame and unique for each message, possess the message priority.

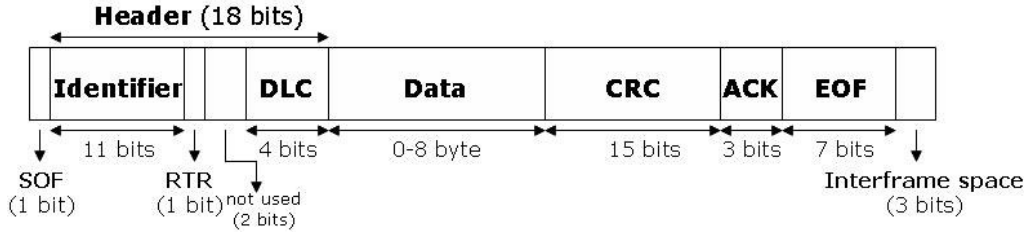


Figure 4.1 CAN 2.0A message format

Each CAN node monitors the bus and when the node detects that the bus is idle, it starts transmission beginning with the identifier field of the message. However, it is possible that other nodes in the network may start transmission at the same time and only one node would continue sending message. The winner node that will complete transmission without any pre-emption is decided based on the CAN arbitration procedure that lasts for the length of the identifier field. Since the CAN bus operates as an AND operator (also OR operator is possible), "0" is the dominant bit on the bus so that the message with the identifier field that is the least in value is granted to be transmitted while other ready messages have to wait. When a node, monitoring the bus, detects a signal with the same polarity (0 or 1) as the one that the node has just sent, it continues to transmit the message; otherwise it immediately stops transmission and waits for another idle period of bus. The node that monitors bits on the bus with the

same as the identifier bits of the message, it is sending, wins the arbitration. Once a message wins arbitration, pre-emption of the ongoing transmission is not allowed.

Since the bus arbitration is based on message priorities, priority scheduling plays an important role in communication performance. There are some scheduling policies to define the priorities of CAN message frames. The scheduling policies that can be applied over CAN network can be classified in two groups: fixed (static) and dynamic algorithms. In fixed priority scheduling, the identifiers of messages are designated according to periods (Rate Monotonic, RM) or deadlines (Deadline Monotonic, DM). Priority designation is performed offline (before system run) and the identifiers of messages do not change during the arbitration phases. References [49][50][51] discuss fixed priority scheduling of messages on the CAN bus and analyze worst case message response times to determine schedulability (response time of a message instance should be less than deadline). In [52], also schedulability analysis of the CAN messages with fixed priorities is discussed including error models and it is shown that existing worst case response time analysis is optimistic especially under high network loads.

In [53][54][55][56] it is shown that dynamic scheduling algorithms perform better by achieving a greater percentage of schedulable message sets especially under high network loads. The Earliest Deadline First (EDF) algorithm is the main representation of dynamic scheduling policies. Because of the high computational overhead and the limited number of identifier bits, approximated EDF scheduling algorithms are applied over CAN in these studies with the aim of a higher degree of schedulability and lower priority inversion. Fixed priority scheduling has the advantage of possessing low computational overhead for host processors in nodes since it is simple and there is no need for a priority update during arbitration phases. However, dynamic scheduling algorithms perform better, providing a greater percentage of schedulable sets with different bus utilization factors.

Moreover CAN has a simple error detection and recovery mechanism, during which receiver nodes check the integrity of the sent message by looking at CRC part of the message. Upon detecting an error, the nodes in the network are informed by error flag messages. Then the message under scrutiny reenters the next arbitration phase to be retransmitted. Approximate error recovery time varies between 17 and 31 bit times. In addition, the CAN protocol provides a fault confinement mechanism by taking the node that exceeds its own error counter to the bus-off state until the counter is reset.

As a result, CAN networks have significant advantages due to event-triggered behavior such as flexibility in efficient bandwidth utilization and response time in addition to easy system extensibility (since there is no static communication schedule). However, error detection/correction and fault confinement mechanisms (i.e. acknowledgment, CRC and automatic retransmission of an erroneous message) provide dependability to some extent since the protocol does not offer additional fault tolerant mechanisms such as bus guardian and membership services. Bus guardian is the component that prevents a node from transmitting outside the protocol specification or its assigned time. Membership service provides the nodes with the knowledge of the set of network nodes performing properly. Different from time-triggered networks, in CAN there is no communication schedule, not providing bandwidth reservation for message frames in the network. Because of this especially under heavy traffic conditions, it is highly possible that low priority messages can suffer from high transmission delays with the difficulty in verifying the delay bound under worst case requirements [5]. This causes lack of predictability and composability in temporal manner.

The CAN protocol has some drawbacks considering fault detection and fault confinement mechanisms. Automatic retransmission of messages following the error flags in the case of corrupted frame detection engages the bus and so induces transmission delay for other messages within the network. In addition, CAN has the "*babbling idiot*" problem [1][25], in which a faulty node repeatedly sends high priority

messages, blocking the bus. In such cases, the node has to do a self-diagnosis, but this may result in non-detection of faults especially caused by logical errors. Thus, additional fault detection and confinement mechanisms are required to make CAN more dependable, which is necessary for safety-critical applications.

4.1.3. Byteflight

Byteflight has been developed by BMW. Byteflight, offering 10 Mb/s, has mainly been used in highly safety related networks (i.e. passive safety) both in automotive and avionic domain that require high bandwidth and dependability. Byteflight is based on the flexible time division multiple access (FTDMA) mechanism, typically using the star network topology. Similar to time-triggered networks, byteflight provides bandwidth reservation for nodes in the network while not using a static, predefined communication schedule. Instead, each node in a byteflight network contains a slot counter that is initiated from "0" upon each synchronization pulse (SYNC). This pulse is sent by a SYNC master node. Similar to CAN, each message exchanged in the network possesses a unique identifier (8 bit) to avoid collision on the bus. Nodes increase their slot counters by 1 upon detecting a mini slot that is seen on the bus after each message transmission. Then the message with the identifier equal to the slot counter value is transmitted by the respective node. If the transmission does not start during a predefined small time interval (after each mini-slot), a successive mini-slot is detected and slot counters are increased again. This procedure continues until the new SYNC pulse by which nodes reset their slot counters to 0.

Providing temporal isolation between messages makes the protocol have a higher degree of dependability compared to event-triggered approaches. The babbling idiot problem can be masked using a star coupler [10]. And finally, since communication in byteflight does not depend on a schedule, it can be considered more extensible than time-triggered networks. From a communication point of view, extending a byteflight

network requires only updating message identifiers based on the message transmissions to be performed.

4.1.4. Time-Triggered Protocol (TTP/C)

The TTP/C protocol is based on the TDMA mechanism offering a bus speed of up to 25 Mb/s. The communication is performed based on a static, predefined communication schedule that consists of cyclically repeated TDMA rounds. In TTP/C, TDMA rounds are partitioned to time slots that are not necessarily equal in duration. Bandwidth reservation is implemented by assigning the time slots to respective nodes. In a TTP/C network temporal isolation is provided by allowing a node to access to the bus only during the time slot that is reserved for that node. The duration of slots in the same TDMA round does not need to be equal, but the duration of a slot in a round is constant and does not change in other TDMA rounds. The communication schedule is stored in communication controllers of each node as a message descriptor list (MEDL). Time synchronization is provided such that the messages in the network are transmitted on a global time base. TTP frames contain 240 B of data and 4 B of overhead.

The fact that the TTP/C protocol depends on a communication schedule makes it less flexible and less extensible. However, time-triggered behavior allows TTP/C to be predictable and composable in temporal manner [9]. In addition to replicated communication channels/nodes and CRC, fault/error confinement and error handling strategies make the protocol highly dependable and fault tolerant. Bus guardians, membership functions, clique avoidance algorithms and error containment mechanisms for control and data errors are the main strategies for a dependable TTP/C network. Although, these properties make TTP/C more complex and lead to higher costs, they make it suitable for x-by-wire and avionics safety systems.

4.1.5. Time-Triggered Controller Area Network (TTCAN)

The TTCAN communication protocol was developed as a time-triggered version of CAN by Robert Bosch GmbH. TTCAN is implemented as an additional layer on CAN physical and data link layers. It uses the same standards and message formats as CAN. TTCAN is a TDMA based, time-synchronous and cyclic bus protocol, which has slots reserved for particular message transmissions. In contrast to CAN, a TTCAN network has a master node that provides time synchronization among nodes by sending a periodic *reference message* that establishes the cycle-based operation. Moreover, each node in a TTCAN network has its own local clock that works in *network time unit* (NTU). Time synchronization between the nodes in the network is crucial for time-triggered scheduling operations. TTCAN time synchronization can be implemented on two levels: *level* 1 and *level* 2 which is the extension of level 1. Level 1 satisfies minimum necessary requirements for time-triggered communication scheduling for synchronization of nodes, whereas, for level 2 the reference message additionally involves global time information (from the clock of the master node) with high precision (in 2 bytes) in addition to the information provided by level 1.

Similar to TTP, the communication in TTCAN is based on pre-computed and fixed schedule called TTCAN *System Matrix* (SM) that repeats itself cyclically during system run. The SM has a column oriented structure and it consists of rows and columns, which form time windows. The rows in the SM are called *basic cycles* that follow each other. In contrast to TTP, event-triggered traffic is also supported during *arbitration windows* in the TTCAN network. During these windows, bus access is performed based on standard CAN arbitration. Each node in a TTCAN network possesses the temporal information (basic cycle and column number and period) only about the time-triggered messages that it is expected to send or receive.

Since the communication in the TTCAN network depends on SM, the structure of the SM plays an important role on real-time communication performance. There are

numerous studies [57][58][59][60][61][62][63] on SM design for better real-time performance while satisfying the protocol constraints. Depending on a static schedule for the time-triggered communication makes the TTCAN network predictable. Furthermore, in the TTCAN network retransmission of erroneous messages is not allowed as well as nodes are not allowed to transmit messages out of their reserved intervals defined by the SM. By this way, not only temporal isolation is achieved but also "babbling idiot" problem is avoided. Apart from retransmission, TTCAN uses fault tolerance mechanisms offered by CAN. Thus, additional functions, to be implemented on upper layers, are necessary to achieve a higher degree of dependability. Finally, co-existence of both time and event-triggered traffic in the network increases the flexibility.

4.1.6. FlexRay

FlexRay has been developed by a consortium of big automobile companies with the aim of having a high-speed and both dependable and flexible in-vehicle communication protocol. The first protocol specification was published in 2004. FlexRay is based on the TDMA and FTDMA mechanisms with comprising both event-triggered and time-triggered communication. FlexRay can offer bit rates up to 10 Mb/s as bus, star and multiple star network topologies. Messages exchanged in the network contain 254 B of data with 5 B of header. Similar to TTP the communication in a FlexRay network communication is performed based on a static predefined schedule called elementary cycle. This is a one-row cycle that repeats itself cyclically during system run. Other than in TTP, the elementary cycle consists of two main windows: static (time-triggered traffic, TDMA) and dynamic (event-triggered traffic, FTDMA). The static window consists of equal-length slots assigned to nodes. Another difference compared to TTP is that nodes in a FlexRay network may have more than one slot in the static segment of an elementary cycle, which increases flexibility. In the dynamic window that follows the static segment, minislots are assigned to nodes based on message identifiers.

Similar to byteflight, the bus access is performed based on message identifiers and slot counters during the dynamic window. The unused minislots are wasted. Similar to TTCAN, each node has its own local time information (elementary cycle and slot number) when to start transmission or reception about the messages it is expected to transmit/receive. Recent research [64][65] on FlexRay timing analysis and communication schedule construction (assigning slots to nodes and defining slot durations in the elementary cycle) aims to improve real-time performance (less response delay and jitter – bus access optimization) with guaranteed schedulability.

FlexRay supports dual channels (both provides redundancy and higher bandwidth) and provides CRC, bus guardian and clock synchronization strategies. Since the protocol does not provide membership, acknowledgement and mode management services, they should be implemented in higher layers. Yet, existing mechanisms such as CRC, bus guardians, never-give-up (NGU) strategy (strategy of nodes to get into the safe mode after a transient fault) of nodes, trigger monitoring and dual channel redundancy make the protocol enough dependable for safety-critical applications. Moreover, depending on static schedule for the time-triggered communication makes the FlexRay network predictable. In spite of the static communication schedule, existence of event-triggered traffic makes the protocol have some flexibility. FlexRay is seen as a strong candidate for safety-related systems and it is expected to be a de-facto standard for future high-speed automotive applications such as x-by-wire.

4.1.7. Media Oriented Systems Transport (MOST)

MOST [23] was developed to provide in-vehicle multimedia and infotainment systems with communication during the transmission of audio, video, data and control information. The MOST cooperation, a consortium of car makers, system architects and key component suppliers, started to develop a multimedia network in 1998, and now MOST is the de-facto standard for such applications [1]. MOST offers a cost-effective and data-efficient communication infrastructure to interconnect multimedia and

infotainment devices such as GPS navigation, video display, radio, active speakers etc. with a data rate of 25 Mbps. MOST is a synchronous network and it uses point-to-point data transfer, supporting both synchronous and asynchronous traffic. In addition, it uses a master/slave mechanism to synchronize nodes in time and to establish connection between a sender and receiver. MOST employs plastic optical fiber (POF) as the physical layer and it is superior to classical copper wires in providing better resilience to EMI and higher data rates.

In summary, some in-vehicle real-time networks were explained briefly under this section. As being wired in-vehicle communication protocols, they were developed for embedded networks apart from the MOST protocol. The MOST protocol has been designed for multimedia and infotainment communication. Thus, to provide a clear overview Table 4.1 summarizes some properties of the in-vehicle protocols explained previously.

Table 4.1 Summary of wired in-vehicle communication networks

	General	Class	Network bandwidth	Network topology	Scheduling	Fault-tolerance and additional services	Synchronization	Functional domains
LIN	- low-speed - low-cost - time-triggered	Class A	20 kb/s	- bus	- master/slave - polling list based on schedule table	- collision resolution by master node - bandwidth and energy saving services	Synchronization of nodes by the 'Sync' field in a LIN message frame sent by master node	- Body/comfort
CAN	- low-cost, simple - twisted pair - event-triggered - de-facto standard - most widely used	Class B Class C	Up to 1Mb/s	- bus - star	- CSMA/CR - Bitwise arbitration based on message identifiers	- CRC - automatic retransmission - error counter and bus-off state schemes - Additional fault tolerance services are necessary on upper layers	Bit synchronization	- Body/comfort - Powertrain - Chassis
Byteflight	- hybrid paradigm - POF	Class D	10 Mb/s	- star	- FTDMA based on message identifiers - master/slave (for synchronization)	- star coupler (to avoid 'babbling idiot') - CRC	Synchronization of nodes by the 'synch pulse' sent by master node	- Passive safety - Safety-critical applications
TTP/C	- twisted pair or POF - time-triggered	Class D	Up to 25 Mb/s (depends on network topology)	- bus - star	- TDMA - predefined and fixed communication schedule (MEDL)	- replicated channels/nodes - star coupler (star topology) - CRC - bus guardian - membership function - clique avoidance algorithm - error containment mechanisms - never-give-up (NGU) strategy - mode change management (different schedules)	Distributed clock synchronization	- x-by-wire - Chassis (active safety)
TTCAN	- low-cost, simple - twisted pair - hybrid paradigm - time-triggered layer on CAN	Class C	Up to 1Mb/s	- bus - star	- TDMA in exclusive windows, CSMA/CR in arbitration windows - predefined and fixed communication schedule (system matrix) - master/slave (for synchronization)	- CRC - mode change (CAN to TTCAN and vise versa) by master node - Additional fault tolerance services are necessary on upper layers	*Level* 1 and *Level* 2 time synchronization by *reference message* sent by master node	- powertrain - chassis - x-by-wire - safety-critical applications
FlexRay	- hybrid paradigm - twisted pair (bus) or POF (star) - future de-facto standard - can be used in two modes (time or event-triggered)	Class D	10 Mb/s	- bus - star - multi-star	- TDMA in the static segment, FTDMA in the dynamic segment - predefined and fixed communication schedule (elementary cycle) - master/slave (for synchronization)	- scalable dependability [1] - dual channel redundancy (optional) - CRC - never-give-up (NGU) strategy - bus guardians for only time-triggered traffic (optional) - trigger monitoring	Distributed clock synchronization	- powertrain - chassis (active safety) - x-by-wire
MOST	- cost-effective - data-efficient - hybrid paradigm - de-facto standard for multimedia/infotainment - POF	Class D	25 Mb/s	- ring - star	- master/slave - support for (a)synchronous - point-to-point video and audio data transfer	- support for "plug and play" - support for multiple master nodes	master node based synchronization by sending the preamble	- multimedia - infotainment

4.2. Middleware Layer

Developments of new applications in automotive embedded systems result in the need for tight cooperation between functions. This requires closer interaction between different networks and tight cooperation between functions. An MW layer is used to facilitate the integration of different software components. The following figure illustrates the layered architecture of an ECU consisting of application (application software components), OS/Middleware and physical layers.

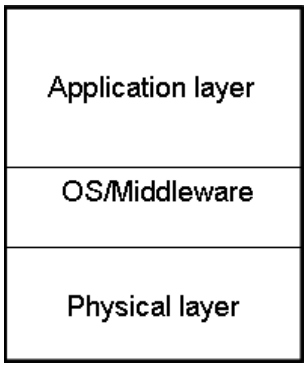


Figure 4.2 Layered software architecture of an ECU

The functions of middleware can be summarized as follows [1]:

- Providing the same communication services for in-node, in-network and inter-network independent from the communication protocols and location of nodes,
- Providing common OS services and an application programming interface with hiding the heterogeneity of communication protocols, and node architectures in the networks,
- Increasing communication quality and reducing development time, providing high level validated services such as membership services, redundancy management, remote procedure call and working mode management etc.,
- Providing frame packing functions,

- Improving QoS provided by communication protocols with additional mechanisms and services such as additional CRC, reliable acknowledgement service and filtering mechanisms.

The main middleware layer examples are OSEK/VDX [66], Volcano [67] and OSEK/VDX FTCom [68]. Moreover, recently AUTOSAR [69] (aims to develop a standard for an automotive software architecture) also proposes a communication layer, providing specifications of the basic software modules and the operating system.

Middleware is designed to support specific in-vehicle network(s). For instance, Volcano was designed to support CAN but later it was extended with services to support also the FlexRay and MOST protocols. Similarly, OSEK/VDX FTCom is an extended version of OSEK/VDX to support the FlexRay and TTP/C protocols, and finally AUTOSAR supports the FlexRay, LIN and CAN protocols.

4.3. Summary

In summary, several wired in-vehicle communication protocols were summarized in this section and they were compared according to defined issues. Based on the following references [9][70][71][72], different in-vehicle protocols were discussed and compared in detail considering protocol specifications. Lastly, the middleware layer concept was discussed briefly.

5. SUMMARY AND CONCLUSION

In summary, this technical report gives a literature survey on in-vehicle communication networks. At first performance attributes that are widely used in the literature for in-vehicle networks' real-time communication are presented. They are used for comparing

different real-time communication paradigms and in-vehicle protocols. In addition, main automotive domains and their requirements are explained in relation to defined performance attributes, and a classification of automotive domains is given based on functions they serve and data rate requirements. Secondly, a common model for in-vehicle network architecture and the structure of components are explained. Moreover, the two most well-known communication paradigms, time-triggered and event-triggered, have been compared using designated attributes. Thirdly, in-vehicle communication protocols for embedded automotive real-time systems, especially used in powertrain, chassis and body domains, are described and compared considering basic properties and mechanisms of the protocols.

The greater number of embedded automotive electronic systems and their functions reveals the increasingly higher complexity of in-vehicle networks. By the nature of in-vehicle environment, automotive electronic systems consist of heterogeneous real-time embedded networks, performing communication between nodes using different in-vehicle communication protocols. Heterogeneity between the node structures (processing unit, memory and I/O interface) may occur even within the same network. At the same time, the demands on performance, reliability, cost and time-to-market are getting tighter. Thus, the design of such systems is becoming more important and difficult resulting in the need for new analysis and design mechanisms for both hardware and software architectures of automotive systems. Analyzing the system requirements helps to define the communication protocol(s) to be used in the networks. Moreover, time analysis (schedulability analysis) becomes a crucial stage for the design and implementation of such systems that have strict timing requirements. Differing system properties and requirements reveal both the component and communication heterogeneity in a vehicle. Because of this, providing communication issue becomes more complex. Therefore providing integrity and interoperability between in-vehicle networks and their components is an important and necessary requirement for dependable and efficient real-time communication.

ACKNOWLEDGEMENTS

I would like to thank R.J. Bril, J.J. Lukkien and M. Holenderski for their valuable feedback on earlier versions of this document.

REFERENCES

[1] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert. Trends in automotive communication systems. *Proceedings of the IEEE*, Vol. 93, No. 6, pp. 1204-1224, June 2005.

[2] A. Albert. Comparison of event-triggered and time-triggered concepts with regard to distributed control systems. *Embedded World*, 17, pp. 235-252, Nürnberg, February 2004.

[3] J. Leohold. Communication requirements for automotive systems. Keynote Presentation. *WFCS 2004, 5th IEEE Workshop on Factory Communication Systems.* September 22-24, 2004.

[4] G. Leen and D. Haffernan. Expanding automotive electronic systems. *In-Vehicle Networks, IEEE*, pp. 88-93, January 2002.

[5] L. Almeida, P. Pedreiras, J. Alberto and G. Fonseca. The FTT-CAN protocol: why and how. *IEEE Transactions on Industrial Electronics*, Vol. 49, No. 6, pp. 1189-1201, December 2002.

[6] I. Broster. Flexibility in dependable real-time communication. PhD Thesis, Department of Computer Science, University of York, August 2003.

[7] N. Navet (editor) and F. Simonot-Lion (editor). Automotive embedded systems handbook. Industrial Information Technology Series. CRC Press, 2009.

[8] A. Avizienis, J.C. Laprie and B. Randell. Fundamental concepts of dependability. LAAS Report no. 01-145.

[9] H. Kopetz. A comparison of CAN and TTP. *Annual Reviews in Control*, No. 24, pp. 177-188, 2000.

[10] T. Nolte. Share-driven scheduling of embedded networks. PhD Thesis. *Malardalen University Press Dissertations*, No. 26. Department of Computer Science and Electronics, Malardalen University, Sweden, May 2006.

[11] N.R. Trevett. X-by-wire, new technologies for 42V bus automobile of the future. Thesis, South Carolina Honors College, April 2002.

[12] F. Simonot-Lion. In car embedded electronic architectures: How to ensure their safety. *5th IFAC International Conference Fieldbus Systems and Their Applications (FeT 2003)*, pp. 1-8, 2003.

[13] E. Dilger, T. Führer, and B. Müller. Distributed fault tolerant and safety critical applications in vehicles – A time-triggered approach. *SAFECOMP'98*, pp. 267-283, 1998.

[14] E. Dilger, T. Führer, B. Müller and S. Poledna: The x-by-wire concept: Time-triggered information exchange and fail silence support by new system services, SAE Technical Paper Series, 980555, February 1998.

[15] H.-G. Frischkorn. Automotive architecture requirements. In *Proceedings of the Summer School "Architectural Paradigms for Dependable Embedded Systems"*, pp. 45-74, Vienna, Austria, September 2005.

[16] J. Leohold. Automotive system architecture. In *Proceedings of the Summer School "Architectural Paradigms for Dependable Embedded Systems"*, pp. 545-591, Vienna, Austria, September 2005.

[17] R. Boys. Safe-by-wire: The leading edge in airbag control. *SAE International*, 2004-01-0205, 2004.

[18] Class C application requirement considerations. Society for Automotive Engineers, Tech. Rep. J2056/1, 1993.

[19] Intel Corp.. Introduction to in-vehicle. [Online]. Available: http://support.intel.com/design/auto/autolxbk.htm, January 2009.

[20] LIN specification package, revision 2.0. [Online]. Available: http://www.lin-subbus.org, July 2008.

[21] Class B data communication network interface. Society for Automotive Engineers J1850 Standard, 1996.

[22] CAN in automation. [Online]. Available: http://www.can-cia.org, June 2008.

[23] MOST specification rev. 3.0. [Online]. Available: http://www.mostnet.de, November, 2008.

[24] D2B/SMARTwireX technology overview. [Online]. Available: http://www.candc.co.uk, December 2008.

[25] G. Cena, A. Valenzano and S. Vitturi. Advances in automotive digital communications. *Computer Standards & Interfaces*, No. 27, pp. 665-678, January 2005.

[26] Time-triggered protocol TTP/C, high level specification document, protocol version 1.1. [Online]. Available: http://www.tttech.com, June 2008.

[27] FlexRay communication system, protocol specification, version 2.0. [Online]. Available: http://www.flexray.com, June 2008.

[28] R. Makowitz and C. Temple. FlexRay – A communication network for automotive control systems. *IEEE*, pp. 207-212, 2006.

[29] J. Berwanger, M. Peller and R. Griessbach. (1999) A new high-performance data bus system for safety-related applications. [Online]. Available: http://www.byteflight.com/specification, June 2008.

[30] J. Berwanger, M. Peller and R. Griessbach. *byteflight*-A new protocol for safety critical applications. *FISITA World Automotive Congress*, Seoul, pp. 1-7, June 2000.

[31] H. Kopetz. Should responsive systems be event-triggered or time-triggered? Research Report Nr. 16/93, November 1993.

[32] H. Kopetz. A comparison of CAN and TTP. *Annual Reviews in Control*, No. 24, pp. 177-188, 2000.

[33] T. Meyerowitz, C. Pinello and A. Sangiovanni-Vincentelli. A tool for describing and evaluating hierarchical real-time bus scheduling policies. *DAC 2003*, pp. 312-317, June 2003.

[34] L. Almeida. Real-time networks for distributed embedded systems: A focus on operational flexibility. *IFAC Summer School*, Control, Computing and Communication, Prague, 2005.

[35] P. Welander. (2008) Fieldbus: Growing globally. [Online]. Available: http://www.controleng.com/article/CA6539014.html, 22 July 2008.

[36] P. Pop, P. Eles, and Z. Peng. Analysis and optimization of heterogeneous real-time embedded systems. *IEE Proc. -Comput. Digit. Tech.*, Vol. 152, No. 2, p.p. 130-147, March 2005.

[37] K. Tindell and J. Clark. Holistic schedulability analysis for distributed hard real-time systems. *Microprocessing and Microprogramming*, Vol. 40, pp. 117-134, 1994.

[38] Road Vehicles-Low speed serial data communication-Part 3: Vehicle area network (VAN). ISO 11 519-3, 1994.

[39] M. Farsi, K. Ratcliff and M. Barbosa. An overview of controller area network. *Networking Systems, Computing & Control Engineering Journal*, pp. 113-120, June 1999.

[40] N. Navet. Controller area network. *IEEE Potentials*, pp. 12-14, 1998.

[41] G. Leen and D. Heffernan. TTCAN: A new time-triggered controller area network. *Microprocessors and Microsystems*, Vol. 26, pp. 77-94, 2002.

[42] A. Albert, R. Strasser and A. Trachtler. Migration from CAN to TTCAN for a distributed control system. *9th International CAN Conference (ICC 2003)*, pp. 5-16, 2003.

[43] T. Fuhrer, B. Muller, W. Dieterle, F. Hartwich, R. Hugel, M. Walther and R. Bosch GmbH. Time triggered communication on CAN (Time Triggered CAN-TTCAN). *Proceedings of 7th International CAN Conference*, pp. 92-98, 2000.

[44] B. Muller, T. Fuhrer, F. Hartwich, R. Hugel, H. Weiler and R. Bosch GmbH. Fault tolerant TTCAN networks. *Proceedings of the 8th International CAN Conference (ICC)*, pp. 2-9, 2002.

[45] F. Hartwich, B. Muller, T. Fuhrer, R. Hugel and R. Bosch GmbH. CAN network with time triggered communication. *Proceedings of the 7th International CAN Conference*, pp. 1-7, 2000.

[46] L. Almeida, J.A. Fonseca and P. Fonseca. Flexible time-triggered communication on a controller area network. *WiP session of RTSS'98*, Madrid, pp. 1-4, December 1998.

[47] P. Pedreiras and L. Almeida. Combining event-triggered and time-triggered traffic in FTT-CAN: Analysis of the asynchronous messaging system. *WFCS-2000*, pp. 67-75, September 2000.

[48] U. Keskin. In-vehicle communication. TU/e SAN Group RTS Regular Meeting Presentation. December 2008.

[49] K.W. Tindell, H. Hansson and A.J. Wellings. Analysing real-time communications: Controller area network (CAN). *Proceedings of Real-Time Systems Symposium*, pp. 259-263, December 1994.

[50] K. Tindell, A. Burns. Guaranteeing message latencies on controller area network (CAN). *Proceedings of the 1st International CAN Conference*, pp. 2-11, September 1994.

[51] K. Tindell, A. Burns and A.J. Wellings. Calculating controller area network (CAN) message response times. *Control Engineering Practice*, Vol. 3, pp. 1163-1169, August 1995.

[52] 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 (2007)*, Vol. 35, pp. 239-272, 2007.

[53] K.M. Zuberi and K.G. Shin. Scheduling messages on controller area network for real-time CIM applications. *IEEE Transactions on Robotics and Automation*, Vol. 13, No. 2, pp. 310-314, April 1997.

[54] K.M. Zuberi and K.G. Shin. Design and implementation of efficient message scheduling for controller area network. *IEEE Transactions on Computers*, Vol. 49, No. 2, pp. 182-188, February 2000.

[55] A. Meschi, M. Di Natale and M. Spuri. Earliest deadline message scheduling with limited priority inversion. *Proceedings of the 4th WPDRTS*, pp. 87-94, 1996.

[56] M. Di Natale. Scheduling the CAN bus with earliest deadline techniques. *Proceedings of the 21st IEEE Real-time Systems Symposium*, pp. 259-268, November 2000.

[57] J. Fonseca, F. Coutinho and J. Barreiros. Scheduling for a TTCAN network with a stochastic optimization algorithm. *International CAN in Automation Conference*, pp. 10-16, 2002.

[58] F. Coutinho, J. Barreiros and J. Fonseca. Scheduling for a TTCAN network with a stochastic optimization algorithm. *4th IFAC FET'2001*, pp. 2-7, November 2001.

[59] A. Albert and R. Hugel. Heuristic scheduling concepts for TTCAN networks. *International CAN in Automation Conference (ICC)*, pp. 01/07-01/09, 2005.

[60] R. Johannson. Time and event triggered communication scheduling for automotive applications. Chalmers Lindholmen University College, Goteborg, Sweden, Tech. Rep. Technical Report 17, 2004.

[61] M. Naughton and D. Heffernan. SMART-Plan: A new message scheduler for real-time control networks. *ISSC 2005*, pp. 302-307, September 2005.

[62] K. Schmidt and E.G. Schmidt. Systematic message schedule construction for time-triggered CAN. *IEEE Transactions on Vehicular Technology*, Vol. 56, No. 2, pp. 3431-3441, November 2007.

[63] U. Keskin. Time-triggered controller area network (TT-CAN) communication scheduling: A systematic approach. MSc. Thesis. The Graduate School of Natural and Applied Sciences, Middle East Technical University, Ankara, Turkey, August 2008.

[64] T. Pop, P. Pop, P. Eles, Z. Peng and A. Andrei. Timing analysis of the FlexRay communication protocol. *Proceedings of the 18th Euromicro Conference on Real-Time Systems (ECRTS'06)*, pp. 203-216, July 2006.

[65] T. Pop, P. Pop, P. Eles and Z. Peng. Bus access optimization for FlexRay-based distributed embedded systems. *2007 EDAA*, pp. 51-56, 2007.

[66] OSEK/VDX Communication, Version 3.0.3. OSEK Consortium. [Online]. Available: http://osek-vdx.org, December 2008.

[67] L. Casparsson, A. Rajnak, K. Tindell and P. Malmberg. Volcano — A revolution in on-board communications. Volvo, Tech. Rep. 98-12-10, 1999.

[68] OSEK/VDX Fault-Tolerant Communication, Version 1.0. OSEK Consortium. [Online]. Available: http://osek-vdx.org, December 2008.

[69] AUTOSAR AUTomotive Open System Architecture. [Online]. Available: http://www.autosar.org/.

[70] F. Ataide, M.M. Santos and F. Vasques. A comparison of the communication impact in CAN and TTP/C networks when supporting steer-by-wire systems. *2004 IEEE International Conference on Industrial Technology (ICIT)*, pp. 1078-1083, 2004.

[71] H. Kopetz. A comparison of TTP/C and FlexRay. TU Wien Research Report 10/2001. May 2001.

[72] A. Albert and W. Gerth. Evaluation and comparison of the real-time performance of CAN and TTCAN. *9th International CAN Conference (ICC 2003)*, pp. 05/01-05/08, 2003.