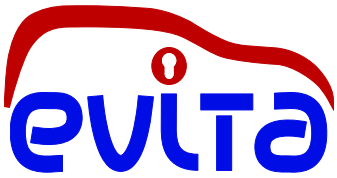

Project acronym:	EVITA
Project title:	E-safety vehicle intrusion protected applications
Project reference:	224275
Programme:	Seventh Research Framework Programme (2007-2013) of the European Community
Objective:	ICT-2007.6.2: ICT for cooperative systems
Contract type:	Collaborative project
Start date of project:	1 July 2008
Duration:	From July 2008 to June 2011 (36 months)

Deliverable D2.1:
Specification and evaluation of e-security relevant use cases

Authors:	Enno Kelling (Continental Teves AG & Co. oHG); Michael Friedewald, Timo Leimbach (Fraunhofer Institute ISI); Marc Menzel, Peter Säger (Continental Teves AG & Co. oHG); Hervé Seudié (Robert Bosch GmbH); Benjamin Weyl (BMW Research and Technology GmbH)
Reviewers:	Alastair Ruddle (MIRA Ltd.); Andreas Fuchs, Olaf Henniger (Fraunhofer Institute SIT); Muhammad Sabir Idrees (EURECOM); Marko Wolf (escrypt GmbH)

Dissemination level:	Public
Deliverable type:	Report
Version:	1.0
Submission date:	4 March 2009

Abstract

The objective of the EVITA project is to design, verify, and prototype a security architecture for automotive on-board networks where security-relevant components are protected against tampering and sensitive data are protected against compromise. Thus, EVITA will provide a basis for the secure deployment of electronic safety applications based on vehicle-to-vehicle and vehicle-to-infrastructure communication. This document describes respective use cases of automotive on-board networks that are expected to require specific security measures. These use cases will serve as a basis for the deduction of security requirements for automotive on-board networks.

Contents

1 Introduction 1
1.1 EVITA objectives 1
1.2 Scope 1
1.3 Document Outline 2

2 State of the Art and Perspectives of On-Board Networks 3
2.1 Communication Architecture 3
2.2 Bus Systems 5
2.3 Reference Architecture for EVITA Use Case Descriptions 11

3 Use Cases 13
3.1 Overview and categories 13
3.2 Use Case – (Car 2 My Car) Safety reaction: Active brake 13
3.3 Use Case – (Car 2 My Car) Local Danger Warning from other Cars 18
3.4 Use Case – (Car 2 My Car) Traffic Information from other Entities 20
3.5 Use Case – (MyCar2Car) Messages lead to safety reaction 22
3.6 Use Case – (My Car 2 Car) Local Danger Warning to other Cars 25
3.7 Use Case – (My car 2 Car) Traffic Information to other Entities 27
3.8 Use Case – (Car2 I and I 2 Car) eTolling 29
3.9 Use Case – (Car2 I and I 2 Car) eCall 32
3.10 Use Case – (Car 2 I and I 2 Car) Remote Car Control 35
3.11 Use Case – (Car 2 I and I 2 Car) Point of Interest 37
3.12 Use Case – (Nomadic Device) Install applications 39
3.13 Use Case – (Nomadic Device) Secure Integration 41
3.14 Use Case – (Nomadic Device) Personalize the car 44
3.15 Use Case – (Aftermarket) Replacement of Engine ECU 46
3.16 Use Case – (Aftermarket) Installation Car2x Unit 48
3.17 Use Case – (Diagnosis) Remote Diagnosis 51
3.18 Use Case – (Diagnosis) Remote Flashing 53
3.19 Use Case – (Diagnosis) Flashing per OBD 56

4 Summary and Outlook 60
4.1 Summary 60
4.2 Outlook 60

References 61

List of figures

Figure 1 Car2X Communication System Architecture 3
Figure 2 Generalised on-board network architecture 4
Figure 3 Logic levels of two-wire CAN bus and fault-tolerant two-wire CAN bus 7
Figure 4 MOST Data Flow (Source: MOST Specification) 9
Figure 5 Internet Protocol as a common interface between applications and different physical networks 10
Figure 6 EVITA Use Cases Reference Architecture 11
Figure 7 Communication Entities and Relations: Safety Reaction: Active Brake 15
Figure 8 Communication Entities and Relations: Local Danger Warning from other Cars 18
Figure 9 Communication Entities and Relations: Traffic Information from other Entities 20
Figure 10 Communication Entities and Relations: Messages Lead to Safety Reaction ... 23
Figure 11 Communication Entities and Relations: Local Danger Warning to other Cars 25
Figure 12 Communication Entities and Relations: Traffic information to other entities . 28
Figure 13 Communication Entities and Relations: eTolling 30
Figure 14 Communication Entities and Relations: eCall 33
Figure 15 Communication Entities and Relations: Remote Car Control 35
Figure 16 Communication Entities and Relations: Point of Interest 38
Figure 17 Communication Entities and Relations: Install Applications 40
Figure 18 Communication Entities and Relations: Secure Integration 42
Figure 19 Communication Entities and Relations: Personalize the Car 44
Figure 20 Communication Entities and Relations: Replacement of Engine ECU 46
Figure 21 Communication Entities and Relations: Installation Car2X Unit 49
Figure 22 Communication Entities and Relations: Remote Diagnosis 52
Figure 23 Communication Entities and Relations: Remote Flashing 54
Figure 24 Communication Entities and Relations: Flashing per OBD 57

List of tables

Table 1 Scenario: Safety Reaction: Active Brake 16
Table 2 Scenario: Local Danger Warning from other Cars 19
Table 3 Scenario: Traffic Information from other Entities 21
Table 4 Scenario: Messages Lead to Safety Reaction 24
Table 5 Scenario: Local Danger Warning to other Cars 26
Table 6 Scenario: Traffic Information to other Entities 28
Table 7 Scenario: eTolling 31
Table 8 Scenario: eCall 34
Table 9 Scenario: Remote Car Control 36
Table 10 Scenario: Point of Interest 38
Table 11 Scenario: Install Applications 40
Table 12 Scenario: Secure Integration 43
Table 13 Scenario: Personalize the Car 45
Table 14 Scenario: Replacement of Engine ECU 47
Table 15 Scenario: Remote Diagnosis 52
Table 16 Scenario: Remote Flashing 55
Table 17 Scenario: Flashing per OBD 58

List of abbreviations

ABS	Anti-Lock Braking System
ACU	Airbag Control Unit
API	Application Programming Interface
BEM	Body Electronic Module
CAN	Controller Area Network
CPU	Central Processing Unit
CSC	Chassis Safety Controller
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CU	Communication unit
DSRC	Digital Short Range Communication
ECU	Electronic Control Unit
EMC	Electromagnetic Compatibility
ESC	Electronic Stability Control
FTDMA	Flexible Time Division Multiple Access
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HU	Head Unit
HMI	Human Machine Interface
IP	Internet Protocol
LDW	Local Danger Warning
LIN	Local Interconnected Network
MOST	Media Oriented Systems Transport
OBD	On-Board Diagnostics
OBU	On-Board Unit
OEM	Original Equipment Manufacturer
PTC	Powertrain Controller
PSAP	Public Safety Access Point
RSU	Road Side Unit
SC	Service Center
TCS	Traction Control System
TDMA	Time Division Multiple Access
UART	Universal Asynchronous Receiver-Transmitter
UMTS	Universal Mobile Telecommunications System
USB	Universal Serial Bus

Document history

Version	Date	Changes
0.1	20/10/08	Initial version
0.2	21/11/08	Document reworked Henniger / Kelling
0.3	20/01/09	Document reworked Saeger / Kelling for WT2100 internal review
0.4	04/02/09	Document reworked Henniger / Kelling after WT2100 internal review
0.5	12/02/09	Minor changes Kelling before and after review at EVITA Meeting 10/02/09
0.6	23/02/09	Final version ready for final editing
1.0	04/03/09	Final version

1 Introduction

1.1 EVITA objectives

The objective of the EVITA project is to design, to verify, and to prototype a modular, (cost-) efficient security solution for automotive on-board networks in order to protect sensitive data within such networks against compromise and, in doing so, to enable secure communication among cars and between cars and infrastructure. By focusing on the protection of the on-board network, EVITA complements other e-safety related projects that focus on the protection of inter-vehicular communication.

The security solution is anticipated to be based on hardware-based security anchors and a software security layer that makes use of these hardware security modules. The solution may apply e.g.

- secure boot,
- secure memory,
- secure security artefacts processing,
- unique ID,
- runtime environment partitioning, and
- secure communication.

1.2 Scope

This report describes use cases of automotive on-board networks that are expected to require security measures. The use cases will later serve as a basis for the deduction of security requirements for automotive on-board networks. The security requirements will in turn serve as input for the design of a secure on-board architecture and the development of appropriate security measures in order to prevent, or at least detect, attacks on automotive on-board networks. However, the security requirements analysis and the design of the secure on-board architecture are out of scope of this report. They will be treated in subsequent EVITA project reports.

For each use case, this report

- defines functionalities required to support the use case,
- defines communication entities (e.g. vehicles, driver, backend infrastructure) and communication relations,
- specifies required data (in-vehicle, backend) as well as exchanged information,
- describes technical requirements (e.g. performance, bandwidth, distance, etc.) without yet considering security.

1.3 Document Outline

The remainder of the report is structured as follows: Section 2 gives an overview of the state of the art and of future trends of automotive bus systems. Section 3 contains the use case descriptions. Section 4 contains concluding remarks.

2 State of the Art and Perspectives of On-Board Networks

2.1 Communication Architecture

2.1.1 Car-to-Car and Car-to-Infrastructure Communication Architecture

The system architecture with the communication entities for car-to-car and car-to-infrastructure (car2X) communication (or, more generally, vehicle-to-vehicle and vehicle-to-infrastructure communication as “vehicle” also includes e.g. motorcycles) is described in the Manifesto Document of the Car 2 Car Communication Consortium [1]. Figure 1 shows a high abstraction level of system architecture as defined in [1].

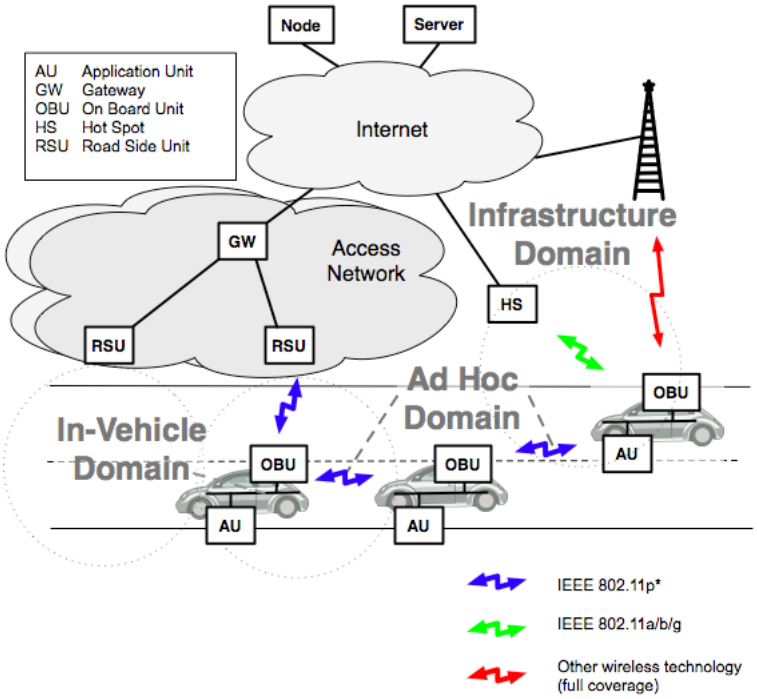

Figure 1 Car2X Communication System Architecture

All of the communication entities depicted in Figure 1 will be taken into account in EVITA. However, the in-vehicle communication system and the communication interface to the outside world are of main interest. Security measures for inter-vehicular communication are e.g. investigated in the European project SEVECOM [2].

2.1.2 On-Board Network Architecture

Automotive on-board networks consist of

– electronic control units (ECUs), comprising a CPU, memory, and I/O devices,

– electronic sensors, and

– electronic actuators

that are connected with each other via some bus systems. The on-board network may possess wireless interfaces to the outside for communicating with service providers, road side units, and other vehicles and a wire-bound diagnostic interface. The on-board network may also possess wireless or wire-bound interfaces for connecting with mobile devices inside the car. The embedded ECUs run both

– safety critical software applications and

– non-safety critical software applications.

Figure 2 shows a generalised on-board network architecture.

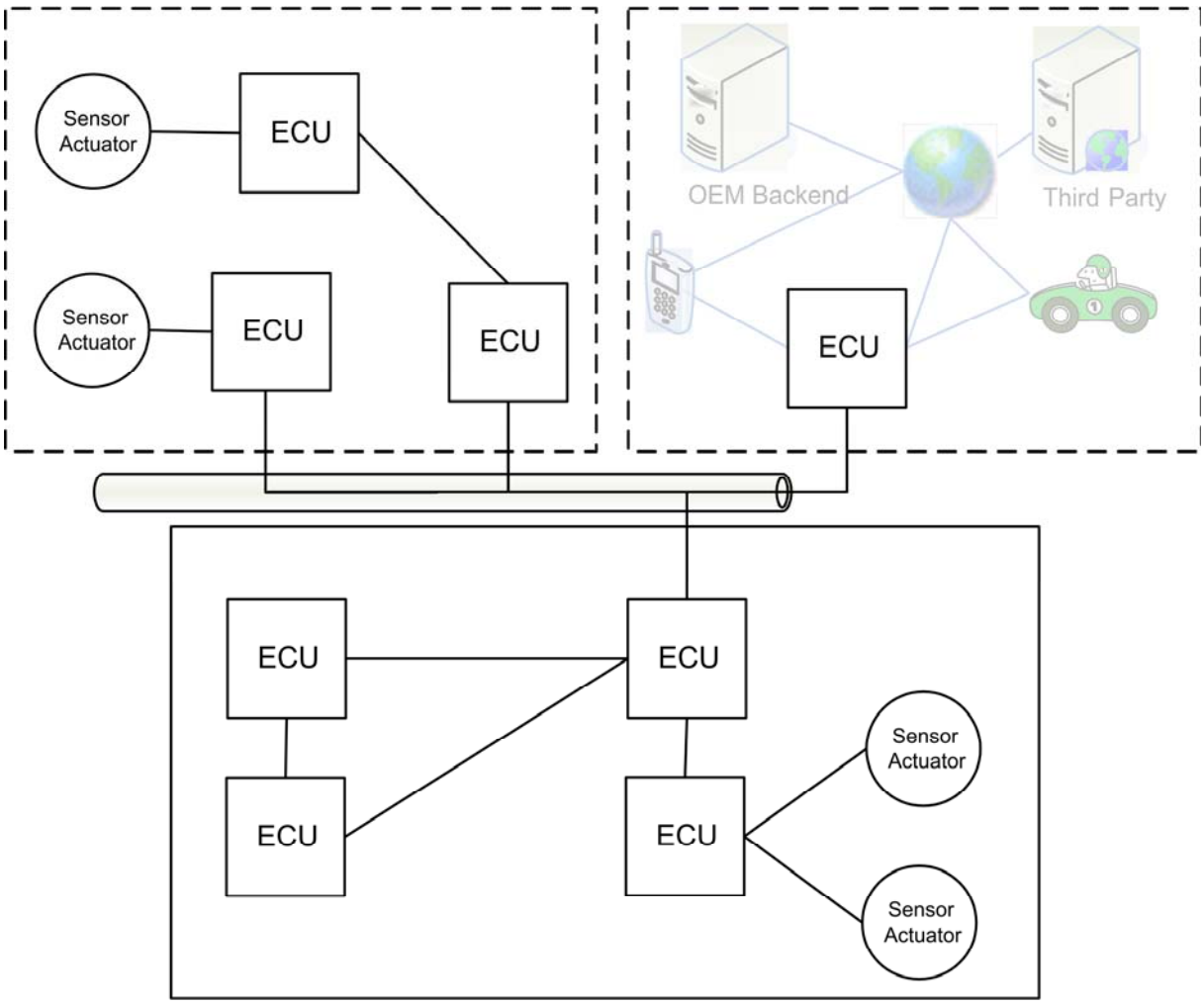

Figure 2 Generalised on-board network architecture

Figure 2 shows that ECUs, sensors and actuators can be clustered in domains and sub-domains that can be interconnected with each other via various communication links. Thereby, different architectural topologies, e.g. line or star topology, are possible.

The on-board network is assumed to operate in an uncontrolled environment. Therefore, its assets must be protected against a variety of threats.

2.2 Bus Systems

2.2.1 Comparison Criteria

Depending on the communication tasks and requirements different network technologies (bus systems) are used for in-vehicle communication. Commonly used bus systems are described in the Deliverable D0.2.4 of the European research project EASIS [3]. A short extract is shown below: Different network technologies are examined here with the following criteria:

- *Flexibility:* Flexibility refers generally to the ability of a system to adapt to new, different or changing requirements. In network technologies flexibility refers to the requirement that the set of messages carried on the bus shall be re-configurable without re-configuring all devices on the bus. For instance, adding or removing devices from the bus or adding a new message to an existing device must not require the alteration of other devices on the bus in order to maintain proper bus operation.
- *Modularity:* Modularity refers to the property of an architecture being composed of modules, i.e. of units that may be handled (implemented, exchanged, etc.) without internal changes. This property requires well-defined interfaces for the modules. Furthermore, for easy handling, a proper tailoring of the modules is necessary.
- *Scalability:* Scalability of a network depends also on the characteristics of a network concerning flexibility and modularity.
- *Fault tolerance:* A fault tolerant system has to provide at least three activities:
  1. *Fault detection*: The system must detect that a particular state combination has resulted, or will result, in a failure.
  2. *Fault isolation* (damage assessment): The parts of the system which have been affected by the failure must be localized.
  3. *Fault recovery*: Set-up a strategy to recover from the failed situation, perhaps in some degraded manner.
- *Bandwidth:* The communication bus shall provide adequate bandwidth for all functions.
- *Communication Method:* Communication method according to the ISO/OSI reference model.

2.2.2 LIN

LIN (Local Interconnected Network) is a cost efficient communication network used for the smart sensors/actuators in vehicles where the bandwidth and versatility of CAN is not required. The LIN Network is characterised by the following aspects:

Flexibility/Scalability/Modularity
A node in a LIN network does not make use of any information about the system configuration, except for the denomination of the master node. Nodes can be added to the LIN network without requiring hardware or software changes in other slave nodes. The size of a LIN network is typically under 12 nodes (though not restricted to this) due to the small number of 64 identifiers and the relatively low transmission speed.

Fault Tolerance
The actions that master and slave tasks undertake upon receipt of a corrupted communication (Fault Tolerance) depend almost entirely on the system requirements and have to be specified in the application layer. The LIN protocol defines only basic errors such as bit error (transmitted signal is different from monitored signal), checksum error, non-responding slave, and no bus activity.

The fault confinement relies mainly on the master node that shall handle as much as possible of error detection and error recovery such as for example the re-scheduling of a message. An acknowledgment procedure for a correctly received message, as known in CAN, is not defined by the LIN protocol.

Errors cannot be directly signalled by slaves but must be polled by the master. Local communication errors at the transmitter can be observed by comparing the outgoing message stream with the monitored message stream. If a slave node detects an inconsistency it saves this as diagnosis information and provides it on request to the master node. All network nodes can detect checksum errors (global error). The identifier and the data fields in a LIN message are error protected by parity and checksum information, respectively.

Bandwidth
The maximum transmission speed of LIN is 20 kbit/s due to requirements for electromagnetic compatibility (EMC) and clock synchronisation.

Communication Method
LIN uses a single-wire serial communications protocol based on the common SCI (UART) byte-word interface.

The medium access in a LIN network is controlled by a master node so that no arbitration or collision management is required in the slave nodes, thus giving a guarantee of the worst-case latency times for signal transmission.

The identifier of a message denotes the content of a message but not the destination. This communication concept enables the exchange of data in various ways: from the master node (using its slave task) to one or more slave nodes, and from one slave node to the master node and/or other slave nodes. It is possible to communicate signals directly from slave to slave without the need for routing through the master node or broadcasting messages from the master to all nodes in a network.

2.2.3 CAN

The Controller Area Network (CAN) is a serial communications protocol that supports distributed real-time control efficiently.

Flexibility/Scalability Modularity
In CAN systems a CAN node does not make use of any information about the system configuration (e.g. station addresses). Nodes can be added to the CAN network without requiring any change in the software or hardware of any node and application layer.

CAN has quite good scalability as sub-bus system; it can be enlarged as long as the bandwidth of CAN is not exceeded (maximal 30 nodes in one network as advised in ISO 11898 – CAN High-Speed).

Fault tolerance

- *Fault tolerance from transfer medium and topology:* Usually CAN employs symmetrical signal transmission with a two-wire line (CAN-H and CAN-L), which is immune to common mode interference. When one wire is damaged, as to the fault-tolerant CAN-Low-Speed-Specification ISO 11519, the failure will be detected from the comparison between the CAN-H and CAN-L signal when a timeout is reached. By means of switch off into an asymmetrical mode, CAN signals can still be transmitted with the other wire and the common ground.

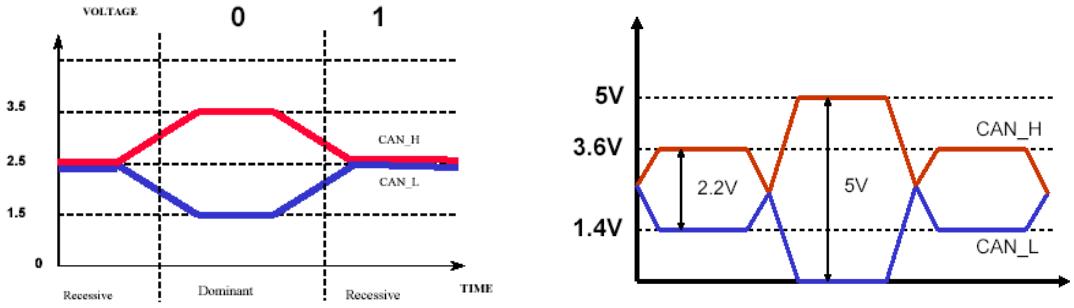

Figure 3 Logic levels of two-wire CAN bus and fault-tolerant two-wire CAN bus

By employment of a double CAN line to connect the nodes, an even better fault tolerance can be reached. In this case, if one of the two-wire lines is broken, the whole network can still work.

- *Fault tolerance from protocol:* An acknowledgment field is implemented in the CAN protocol in which a correctly received message will be sent back from the receiver in the acknowledgment slot.

  The error handling mechanisms of CAN include Error Detection and Error Signalling. For detecting errors the following measures have been taken:

  - Monitoring (transmitters compare the bit levels to be transmitted with the bit levels detected on the bus)
  - Cyclic Redundancy Check
  - Bit Stuffing
  - Message Frame Check

  CAN provides as fault isolation the following services:

  - CAN nodes are able to distinguish short disturbances from permanent failures.
  - Defect nodes can be switched off in case of fault.

Bandwidth

CAN bus up to 500 m provides a bandwidth of 125 kbit/s; up to 40 m also a bandwidth of 1 Mbit/s is possible.

Communication Method

CAN is a multi-master bus with CSMA/CD method. The CAN-messages with different priorities are broadcast to the bus; any number of nodes can receive and simultaneously act upon the same message.

2.2.4 FlexRay

Flexibility/Scalability Modularity
FlexRay requires a so-called communication scheduling, which is not needed in CAN. Because of the TDMA and FTDMA communication methods employed by FlexRay, the designer should define the smallest time slot needed in the system.

FlexRay provides two configurable transfer modes, synchronous and asynchronous mode. Cyclic messages from the control system with real-time requirements can be transferred in the synchronous mode, while event triggered messages or diagnosis messages can be transferred in the asynchronous mode. Due to the flexibility between the dynamic and static segments FlexRay can be used as purely time-triggered or event-triggered. This gives FlexRay more application areas.

Fault Tolerance
FlexRay, compared with the other bus technologies, provides fault tolerance mechanisms not only from the physical layer, like CRC etc., but also directly from the upper protocol layer.

FlexRay provides fault tolerance by the synchronization of the global time and fault tolerance in communication channel (scalable redundant system to improve the dependability). It also provides the mechanism to ensure that the communication system will not be blocked by one node.

FlexRay supports a bus and star topology. In an active-star topology a physical failure like a short-circuit will be found out by the active star node and recovered from by turning off the damaged twig. By means of “Bus Guardian”, as a watchdog in the communication controller, FlexRay can prevent single node failures from producing a ‘babbling idiot’ jamming the bus.

Bandwidth
FlexRay supports a baud-rate of 10 Mbit/s.

Communication Method
FlexRay is a communication system without arbitration and with two communication methods: TDMA (Time Division Multiple Access) and FTDMA (Flexible Time Division Multiple Access). Both methods are characterized by cyclically repeated schema from a mixture of static and dynamic message transmitting (TDMA + FTDMA).

2.2.5 MOST

The Media Oriented Systems Transport (MOST) was designed by Oasis SiliconSystems. MOST is already used by OEMs as a vehicular multimedia bus system in the infotainment domain in vehicles with several multimedia devices like navigation, radio, television, etc.

Flexibility/Scalability Modularity
The MOST specification defines a set of API functions for the multimedia domain, e.g. navigation system, sound system, and so on, which allows from the software point of view a very flexible application development.

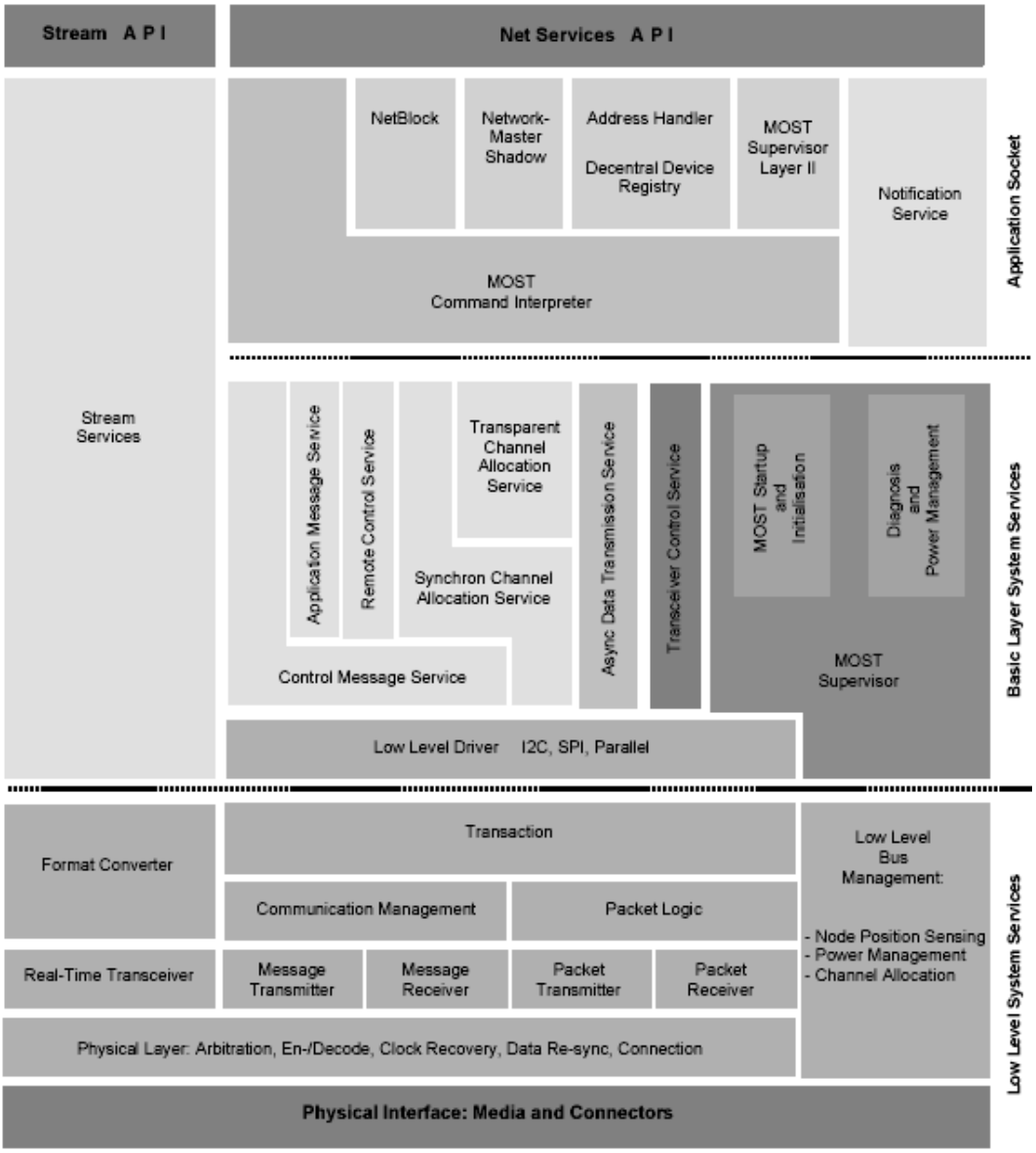

Figure 4 MOST Data Flow (Source: MOST Specification)

Bandwidth
The MOST bus supports a baud-rate of 24.8 Mbit/s.

Communication Method
MOST is a synchronous bus system on the basis of a time master. The basic communication method is master/slave.

The MOST bus has implemented methods to transmit synchronous, asynchronous and control data. Because of the synchronous features low cost multimedia systems are possible. The bit error rate is less than 10^{-10}. Glass or plastic optical fibre can be used as physical transmission media.

2.2.6 New approaches towards IP-based in-car network architecture

During the last years, the number of in-car communication networks has significantly grown due to the deployment of an increasing number of ECUs. Up to 70 different application specific ECUs are deployed in a premium car of the upper class that are connected by different bus systems like CAN, FlexRay, MOST, LIN etc. The network complexity will still grow in the future due to the increasing number of functions. Up to 90% of all innovations in a car are provided or assisted by modern electronic and software. Examples can be found in the area of infotainment and driver assistance systems as well as powertrain or safety systems.

To face the growing complexity and to comply with future demands, new concepts are needed. A radical vision that addresses the described challenges is the use of the Internet Protocol as a common basis for the in-car communication network [10].

The Internet Protocol can be seen as an abstraction layer between the applications and physical network technologies. By using IP, the in-car communication network speaks a common and universal language. The applications can utilize a standardized communication interface independent of the physical network. That means that the network technology (e.g. electrical Ethernet, optical Ethernet, WLAN, Bluetooth, UWB, etc.) can be replaced without the need of changing the application software. This is in principle not the case for current communication systems in the vehicle. Also heterogeneous networks, consisting of different physical transmission technologies, can easily be built by using the Internet Protocol (see Figure 5). They all speak the same ‘language’, namely the Internet Protocol. If the data rate requirements increase from a vehicle generation to the next one so that the current network technology does not suffice any more, it can easily be replaced with a faster one without the need to change the application software. An example is the migration from Fast Ethernet (100 MBit/s) to Gigabit Ethernet in the car’s communication backbone.

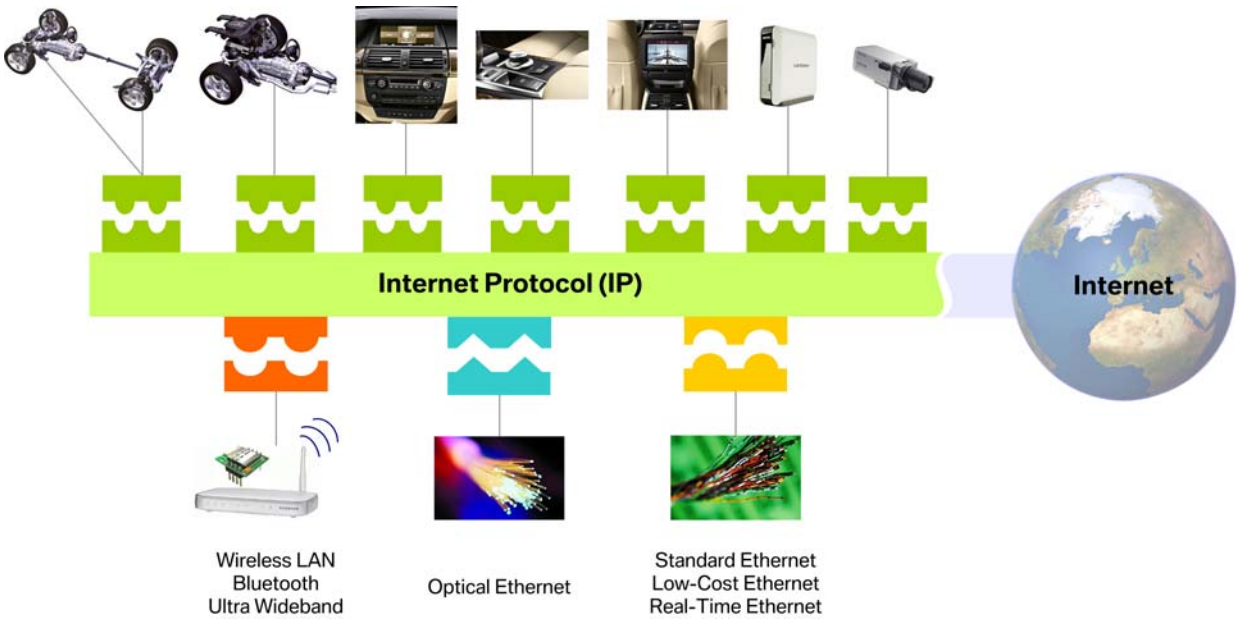

Figure 5 Internet Protocol as a common interface between applications and different physical networks

The described flexibility is emphasized by a modular and standardized software framework which is provided by the IT-world and that is also applicable for the restrictive resource requirements in the automotive world. This modular software framework provides different implementations of the TCP/IP protocol suite with all related protocols. Examples are Universal Plug and Play (UPnP) for controlling consumer electronic devices, the Real-Time Transport Protocol (RTP) for streaming audio and video data or the Simple Network Management Protocol (SNMP) for network management.

The vision of an all-IP in-car communication network entails further advantages regarding the design of the network architecture. By the introduction of IP it would be possible to resign complex gateways since there is no need to convert data and frame formats between completely different network technologies like MOST, CAN, FlexRay or LIN. Thus the gateway is reduced to a simple router of IP packets without the need of processing and converting the data. All this leads to a remarkable complexity reduction in the overall network architecture and to a simplification of the system design and partitioning.

IP has advantages also in terms of the development process. Comparing the different development tools and measurement equipment available for the IP technology with those available for the automotive domain, it becomes clear that the IP technology offers a much wider range at a lower cost. A very prominent example is the open source protocol analyzer “wireshark” which is widely used in the internet area and which is available free of charge.

2.3 Reference Architecture for EVITA Use Case Descriptions

The use case descriptions are based on a common architecture and topology for the in-vehicle communication networks consisting of ECUs, sensors, and actuators. This reference architecture for the use case descriptions is shown in Figure 6.

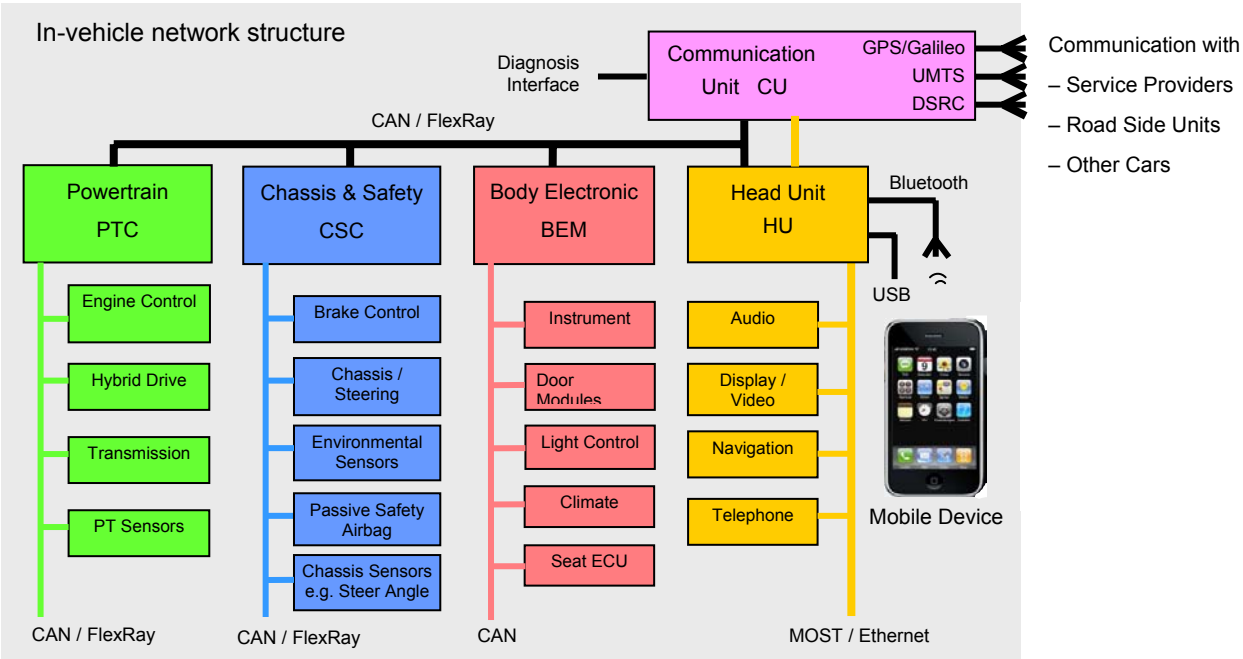

Figure 6 EVITA Use Cases Reference Architecture

Figure 6 shows a generalized topology for on-board networks of contemporary or next-generation high-end cars, but without specific changes according to the requirements of the platforms. The architecture represents an instantiation of the generalised on-board network architecture (cf. Section 2.1.2). This instantiation is used for describing the use cases related to a concrete exemplary in-vehicle network. However, the use cases of course can also be mapped to other instantiations of the reference model.

Within the exemplary instantiation, the control systems are clustered into different domains for Powertrain, Chassis & Safety, Body Electronics and Infotainment (Head Unit). The domain control units provide separate communication networks for their domains, control the high-level domain specific functions and are linked together via a backbone bus system. The communication unit that provides the wireless communication to the outside world on a cellular basis or via digital short-range communication (DSRC) is also connected to the backbone. The Head Unit provides an interface to connect mobile devices to the Infotainment domain e.g. via Bluetooth or USB. With this instantiated architecture use cases can be described and intrusion scenarios can be investigated in order to deduce security requirements.

3 Use Cases

3.1 Overview and categories

Use Cases where an e-safety related intrusion can likely happen are clustered as follows in categories:

– Car2MyCar (communication from other car to own car),

– MyCar2Car (communication from own car to other car),

– Car2I and I2Car (communication from car to infrastructure and from infrastructure to car),

– Nomadic Devices / USB Sticks / MP3,

– Aftermarket,

– Diagnosis.

The use cases are described in terms of the communication relations according to the EVITA use cases reference architecture (cf. Section 2.3). Typical communication sequences are shown in the scenario tables. All specified values (e.g. data length, repetition rate) are assumptions or examples. They may vary among different car types.

3.2 Use Case – (Car 2 My Car) Safety reaction: Active brake

3.2.1 Description

3.2.1.1 General

The car receives a message that indicates that the car is in immediate danger of collision with an object. The only way to avoid the collision is an instant brake manoeuvre.

Remark: The source of the message is described in detail in Section 3.6.

The emergency message contains longitude, latitude and altitude of the dangerous object, the time of message generation, the expiry time of the message, an indicator for the reliability of the information, a code that is classifying the object, an Id that is identifying the sender of the message and an event code that is classifying the emergency situation. All this information is packed in a message frame that adds checksum, information for protocol processing and if necessary security information.

The receiving communication unit (CU) will check the message for correctness and then pass the information together with additional relevant information to the chassis safety controller (CSC). The additional information consists of data about the position, speed, heading, type and size of communicating objects nearby; further attributes may also be added. The additional information was collected from older received messages and stored in the neighbourhood table. The neighbourhood table is a list of communication nodes from which the CU received messages in the past. The list contains the nodes Id, position, type and other available attributes. Nodes that are more than 1 km distant will be deleted from the list.

The information is provided in regular intervals (ca. 2/s). In parallel to the following action, the CU will assess whether it has to send the information out to other nodes. The assessment depends on the position of nearby CUs, the received RF power of the message and the type of the message. Only event messages will be rebroadcast. If it is obvious that all affected units that are even further away from the sender have received the same message, the message will not be rebroadcast, otherwise it is sent out again. The GPS unit of the CU is used to determine the position; this position is used internally and for car2X communication.

The CSC will use further information that is available to perform a plausibility check. This information may be object lists from radar, lidar or video sensors together with data from digital maps, driver status information and status data of the car like position, speed, heading, steering angle etc. Except for the car status all these data sources are optional.

If the plausibility check confirms the danger for the car, the CSC decides on appropriate action, which mainly depends on the possibilities that the vehicle dynamics and the neighbourhood conditions permit. If the CSC decides that a braking manoeuvre is the best solution, it will send a braking command and information concerning the best deceleration to the brake control unit. In addition, information about the emergency-braking manoeuvre will be sent to the CU. The CU will then broadcast an emergency braking message to warn following cars.

The brake control unit (BCU) will adjust the braking mechanics to get a deceleration as close as possible to the desired value, while keeping the car in a controllable state by executing ABS/TCS/ESC algorithms. When starting the braking, the BCU will send a message to the powertrain domain to reduce the driving power. This message is forwarded by the CSC and the Powertrain controller (PTC). The PTC will decide how best to comply with this request and will send the necessary commands to the units of the powertrain domain. The CSC will update the plausibility check and the concluding braking commands in regular intervals (ca. 10/s). The braking commands will be adapted to the situation assessment.

When the CSC gets information from environmental sensors (radar, lidar, video) and/or car internal sensors (digital map, speed, yaw rate, etc.) that show that the dangerous situation is no longer existent or that the driver is fully able and ready to cope with the danger, it returns control to the driver by adapting the deceleration to the braking pedal pressure.

3.2.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 7).

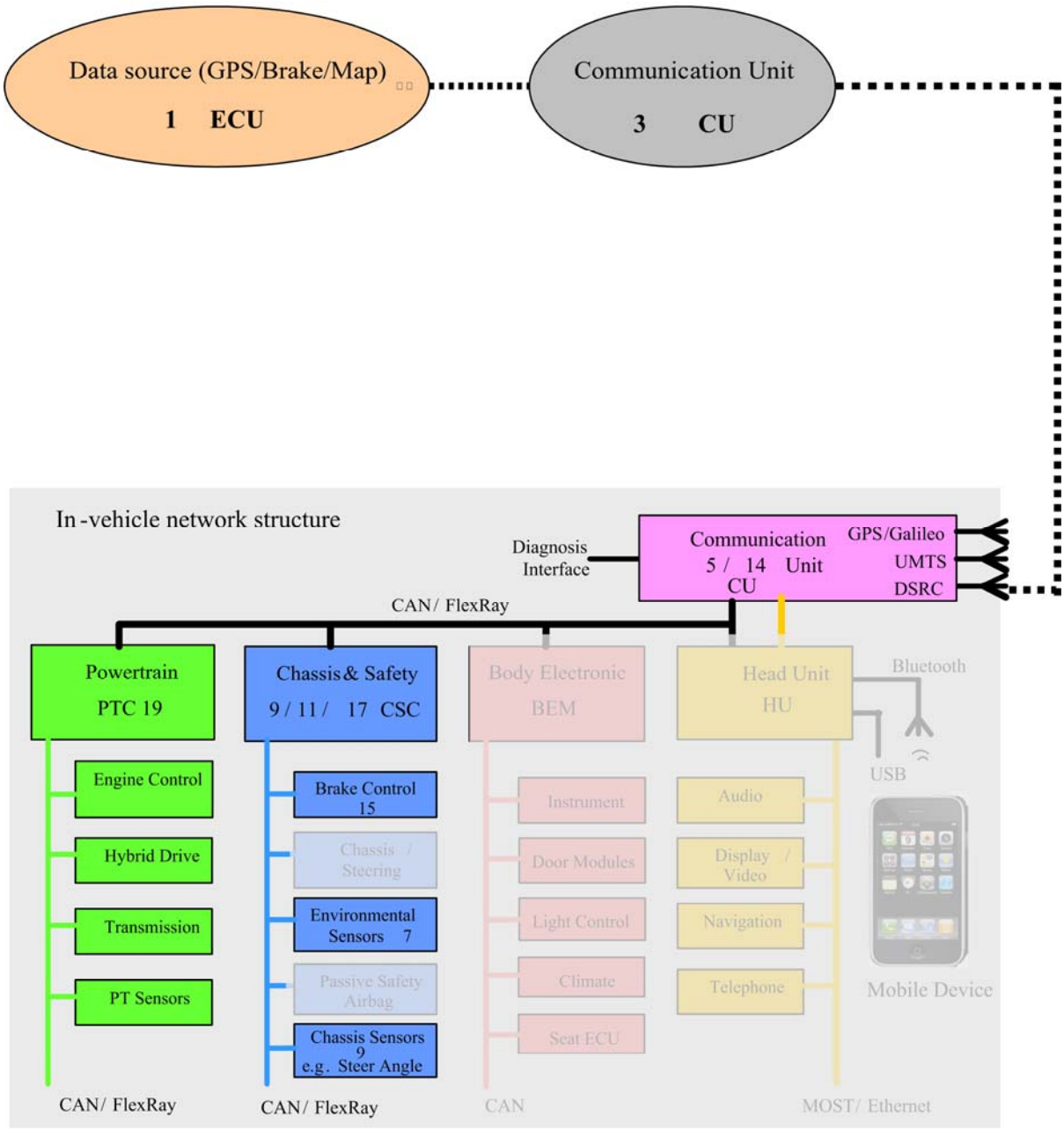

Figure 7 Communication Entities and Relations: Safety Reaction: Active Brake

3.2.2 Scenario

Table 1 Scenario: Safety Reaction: Active Brake

Step no.	Actor	Recipient	Type[1]	Data / act	Data length	Remark
1	ECU	–	algo	Data creation for car2X message		in other car
2	ECU	CU	com	Data sent to CU	500 byte	in other car
3	CU	–	algo	Data packed to message		in other car
4	CU	CU	com	car2X message	500 byte	
5	CU	–	algo	Check message for correctness		
6	CU	CSC	com	Message data + neighbourhood table	1–100 kb	2 Hz repetition + events
7	Environment Sensor	–	algo	Get environment information		In parallel to steps 1-6
8	Environment Sensor	CSC	com	Object list	1–50 kb	In parallel to steps 1-6
9	Chassis Sensor	–	algo	Get vehicle dynamics data	32 byte	In parallel to steps 1-6
10	Chassis Sensor	CSC	com	vehicle dynamics data	32 byte	In parallel to steps 1-6
11	CSC	–	algo	Plausibility check & danger avoidance strategy		10 Hz repetition
12	CSC	BCU	com	Brake command	8 byte	10 Hz repetition
13	CSC	CU	com	Information about emergency braking	8 byte	
14	CU	World	com	Emergency braking	100 byte	
15	BCU	–	algo	Braking		100 Hz repetition
16	BCU	CSC	com	Driving power reduction request	8 byte	100 Hz repetition
17	CSC	–	algo	Bus translation		100 Hz repetition
18	CSC	PTC	com	Driving power reduction request	8 byte	100 Hz repetition
19	PTC	–	algo	Driving power reduction strategy		100 Hz repetition
20	PTC	PT ECUs	com	commands	1–4 × 8 byte	100 Hz repetition

[1] “algo” denotes a process being performed by the actor;
“com” denotes a message transfer from the actor to the recipient.

3.2.3 Requirements

3.2.3.1 Functional requirements

- The car must stay controllable by the driver – no autonomous driving allowed.
- No automatic braking without a real danger is allowed.
- After the emergency brake, a smooth return to driver control has to be secured.
- No failure in any single unit may be succeeded by automatic braking.
- No failure of any single communication may be succeeded by automatic braking.
- Failures that result in a false automatic braking event must not happen more often than 10^{-9}/h (during operation).
- Any single fault in the ECUs has to be detectable.
- Information about dangerous events has to be rebroadcast according to communication congestion control algorithms.
- Information about the emergency braking manoeuvre has to be broadcast to other cars only when an emergency braking occurs.
- For the driver there should be no difference between an emergency braking initialized by an environmental sensor or the car2X communication.

3.2.3.2 Technical Requirements

- The maximum delay from reception of the car2X message to the activation of the brakes should be less than 250 ms.
- Additional security information on the busses in the chassis and powertrain domain should be less than 15% of the net data.

3.2.4 Security Aspects

- The information received from another car needs to be evaluated regarding security and trust (e.g. authenticity of data).
- Privacy of the broadcast car information has to be guaranteed.

3.3 Use Case – (Car 2 My Car) Local Danger Warning from other Cars

3.3.1 Description

3.3.1.1 General

In order to avoid critical driving situations, car2X communication helps to virtually extend the view of the driver. The driver can be warned in critical situations, where an obstacle may be overseen, e.g. intersection warnings based on communication. This kind of application will reduce the number of fatalities.

The communication of local danger warning messages is based on “Cooperative Awareness Messages” CAM and “Decentralized Environmental Notifications” (DEN) specified by the C2C-CC. Within Cooperative Awareness Messages, data is periodically broadcast, e.g. in order to prevent accidents at an intersection. Decentralized environmental notifications provide more specific information about an occurrence, e.g. to indicate a potential danger to other vehicles such as a car with warning lights on.

The car that receives the messages processes the information and provides the driver a corresponding warning.

3.3.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 8).

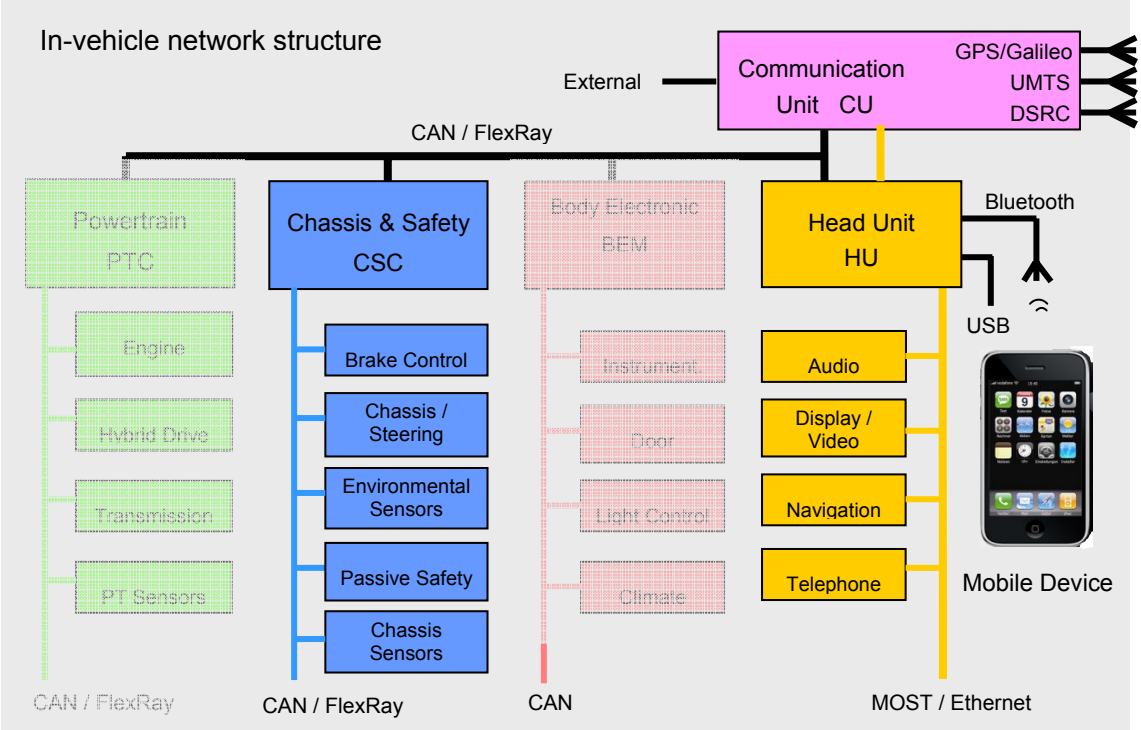

Figure 8 Communication Entities and Relations: Local Danger Warning from other Cars

For local danger warning applications, at least two entities are involved: the car receiving a critical warning message and the entity sending such a message. The entity that sends out the message can be another car, a road-side unit or traffic light, or an infrastructure based-server.

Within the on-board network, the scenario involves the communication unit for sending and receiving messages, the local danger warning application running on a separate ECU or being in coexistence with other applications on the same ECU. Further, a connection to the Human Machine Interface (HMI) is required for displaying the warning message, e.g. via audio signals or on a display. Optionally, local danger warning information can also be provided to in-vehicular safety concepts for further processing.

3.3.2 Scenario

Table 2 Scenario: Local Danger Warning from other Cars

Step no.	Actor	Recipient	Type	Data / act	Data length	Remark
1	Car	–	algo	Recognition of a critical situation relevant for other vehicles.		in other car
2	Car	MyCar	com	Cooperative Awareness Message (CAM)	64–256 Byte	
3	CU	LDW	com	Decapsulation of payload within CAM and forwarding to LDW application		
4	LDW	–	algo	Processing of received information		
5	LDW	HMI/CSC	com	HMI is triggered and relevant information sent to HMI; information can also be provided to a CSC component.		
6	HMI	–	algo	Warning is displayed		
7	CSC	–	algo	Possible processing of information		

3.3.3 Requirements

3.3.3.1 Functional Requirements

– Reliable communication infrastructure between entities

– Application processing received information

– Real-time capabilities of the system

– Possibility to notify the driver

– Optionally, data is provided to in-vehicle safety concepts for further processing.

3.3.3.2 Technical Requirements

Processing time is of major importance within critical situations. Within intersection scenarios a frequency of 10 Hz for sending Cooperative Awareness Messages is required for recognizing and responding appropriately to critical situations.

3.3.4 Security Aspects

Within very critical situations, the used data is processed within a very short time frame. It has to be assured that information received from another entity can be checked to be authentic and trustworthy. Only trustworthy data should be processed by the LDW application. Manipulated or unsecured data should be filtered in advance. The provided information and digital identity of the sending entity needs to be anonymised in order to prevent location tracking.

3.4 Use Case – (Car 2 My Car) Traffic Information from other Entities

3.4.1 Description

3.4.1.1 General

Classic traffic information is limited by high levels of latency and partly by inaccurate information. Enhancing this technology with car2X information from other cars, road-side units and backend service infrastructure will allow more efficient restructuring of traffic flows.

3.4.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 9).

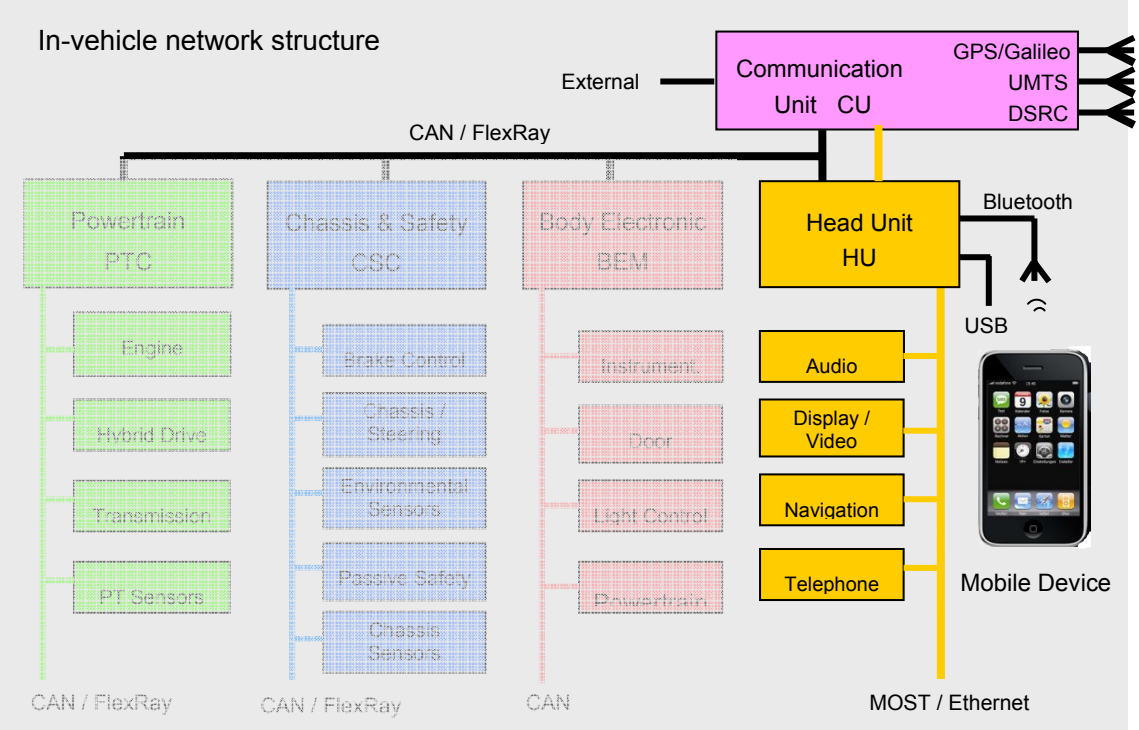

Figure 9 Communication Entities and Relations: Traffic Information from other Entities

The entity that sends out traffic information messages can be another car (cf. Section 3.7), a road-side unit or traffic light, or an infrastructure based-server. Usually, for traffic information, the number of entities providing information enhances accuracy of the data.

Within the on-board network, the scenario involves the communication unit for sending and receiving messages, the traffic information application running on a separate ECU or co-existing with other applications on the same ECU. Further, a connection to the Human Machine Interface (HMI) and the navigation system is required.

3.4.2 Scenario

Table 3 Scenario: Traffic Information from other Entities

Step no.	Actor	Recipient	Type	Data / act	Data length	Remark
1	Car/RSU/ Backend	–	algo	Recognition of relevant traffic information via in-vehicle sensors.		Other car / RSU / Backend
2	Car/RSU/ Backend	MyCar	com	Traffic Information Message	64–256 Byte	
3	CU	Traffic Application	com	Decapsulation of payload and forwarding to application		
4	Traffic Application	–	algo	Processing of received information		
5	Traffic Application	HMI/ Navigation	com	HMI/Navigation is triggered and relevant information sent to HMI.		
6	HMI	–	algo	Warning is displayed		

3.4.3 Requirements

3.4.3.1 Functional Requirements

– Reliable communication infrastructure between entities

– Application that processes the received information

– Possibility to notify the driver

– Respective Backend-Infrastructure in order to aggregate traffic information

3.4.3.2 Technical Requirements

Processing time for traffic information is not as critical as in warning application. However, the “freshness” of the data is important in order to have an up-to-date view on the traffic situation.

3.4.4 Security Aspects

For traffic information, the trustworthiness of the data is of major importance. The recipients have to trust the information.

3.5 Use Case – (MyCar2Car) Messages lead to safety reaction

3.5.1 Description

3.5.1.1 General

When a dangerous situation occurs that forces the driver, or the car itself, to perform a manoeuvre, this can endanger other vehicles. In order to warn other vehicles, the car sends out a warning message. Nearby cars that are in danger can then react according to the information provided within the message.

Remark: The processing of the message in other cars is described in detail in Use Case “Safety reaction: Active brake”.

An ECU of the chassis & safety domain detects a danger; this may be the trigger of an air-bag, an obstacle in direction of travel seen by an environmental sensor, or an emergency braking performed by the driver or an automatic system. The Chassis Safety Controller (CSC) gets information about the dangerous situation via the Chassis Domain Bus. The CSC will assess the situation and will take measures to mitigate the danger for the car. The measures will result in commands to actuator ECUs in the chassis & safety domain and additionally commands to the powertrain domain to get a helpful driving power adjustment. In parallel it will also send information to the Communication Unit (CU). This information will contain data about the current vehicle dynamic status and detailed information about the planned actions (deceleration or acceleration, steering, etc.).

The CU will send out a warning message that contains this information via the DSRC interface to nearby vehicles. The emergency message contains longitude, latitude, altitude, speed, acceleration and heading of the car, the time of message generation, the expiry time of the message, an indicator for the reliability of the information, a code that is classifying the car, an id that is identifying the sender of the message, an event code that is classifying the emergency situation and the planed acceleration and heading. All this information is packed in a message frame that adds checksum, information for protocol processing and if necessary security information.

3.5.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 10).

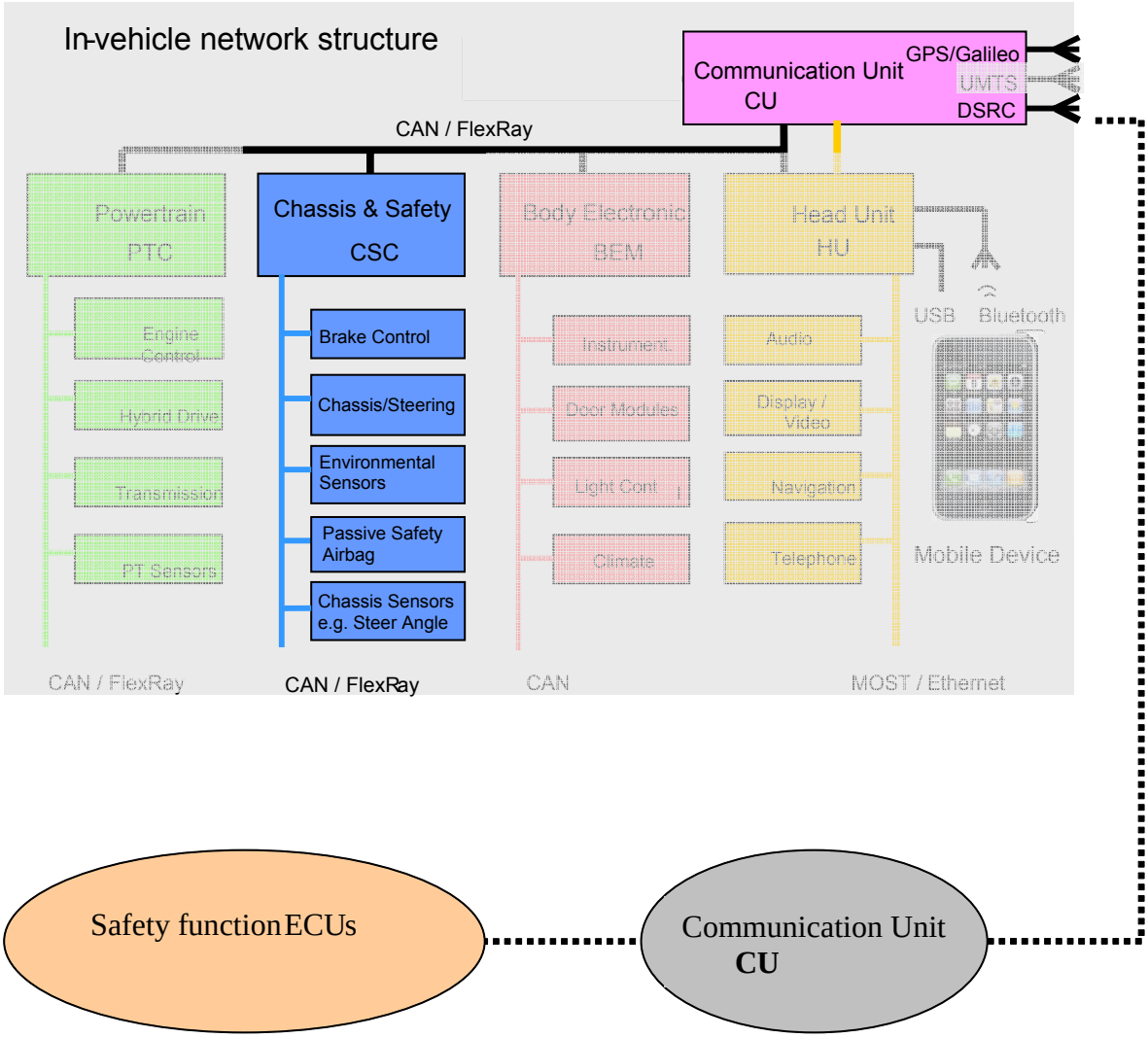

Figure 10 Communication Entities and Relations: Messages Lead to Safety Reaction

3.5.2 Scenario

Table 4 Scenario: Messages Lead to Safety Reaction

Step no.	Actor	Recipient	Type	Data / act	Data length	Remark
1	Chassis & safety domain ECU	–	algo	Danger detection		
2	Chassis & safety domain ECU	CSC	com	Data sent to CU	32 byte	
3	CSC	–	algo	Situation assessment		
4	CSC	Chassis & safety domain ECU	com	Action requests	8–64 byte	100 Hz repetition
5	CSC	CU	com	Data about danger and actions	64 byte	10 Hz repetition
6	CSC	PTC	com	Action requests	16 byte	100 Hz repetition
7	CU	–	algo	Create warning message		
8	CU	CU	com	Warning message	500 byte	2 (5?) Hz repetition
9	CU	–	algo	Check message		Other car or RSU
10	CU	Safety ECU	com	Warning message		

3.5.3 Requirements

3.5.3.1 Functional requirements

– No warning message without a real danger is allowed.

– No failure in any single unit may be succeeded by a false message.

– No failure of any single communication may be succeeded by a false message.

– Any single fault in an ECU has to be detectable.

– Information about dangerous events has to be broadcast according to the communication congestion control algorithms.

– Information about the dangers have to be broadcast to other cars with highest priority.

– Privacy of the broadcast car information has to be guaranteed.

3.5.3.2 Technical Requirements

– The maximum delay from danger detection to broadcast of the car2X message should be less than 150 ms.

– Additional security information on the busses in the chassis & safety and powertrain domains should be less than 15% of the net data.

3.6 Use Case – (My Car 2 Car) Local Danger Warning to other Cars

3.6.1 Description

3.6.1.1 General

In order to avoid critical driving situations, car2X communication helps to virtually extend the view of the driver. The driver can be warned in critical situations, where an obstacle may have been overseen, e.g. intersection warnings based on communication. This use cases describes the detection of a local danger via internal sensors and ECUs of the in-vehicular system, which is used in order to generate a local danger warning message. The local danger warning message is then broadcast to other vehicles. The use case complements the description of the use case in Section 3.3 where the reception of the information is explained.

3.6.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 11).

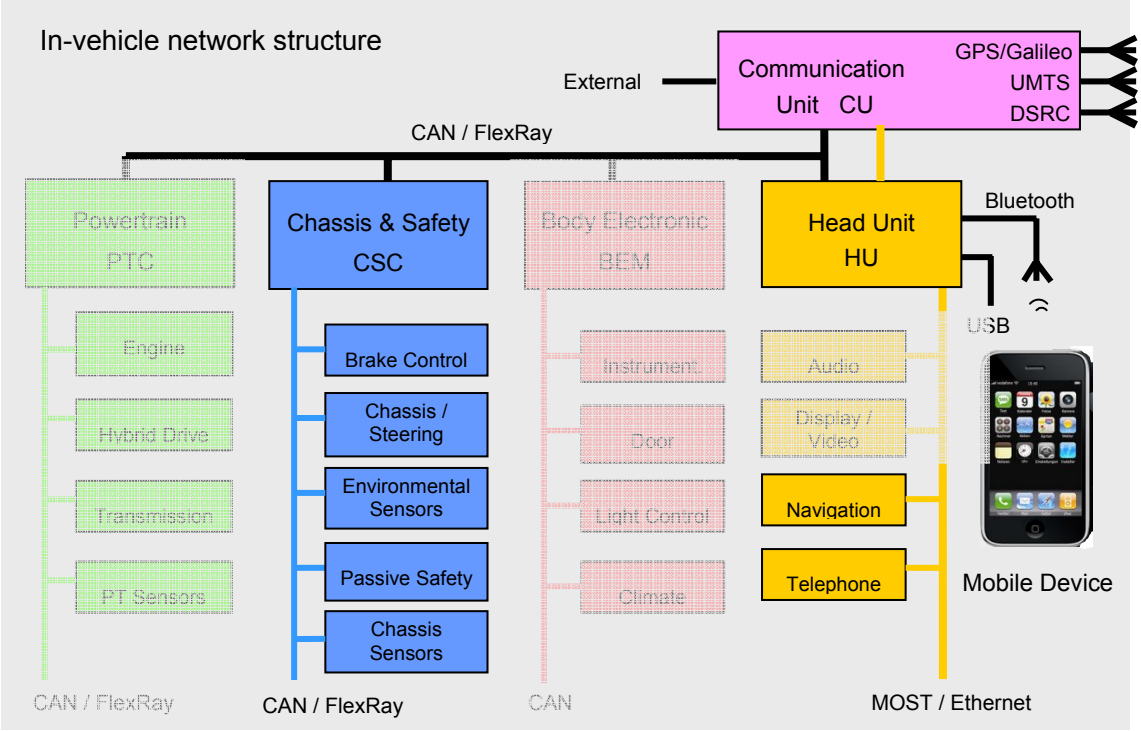

Figure 11 Communication Entities and Relations: Local Danger Warning to other Cars

For local danger warning applications, at least two entities are involved: the car receiving a critical warning message and the entity sending such a message.

Within the on-board network, the scenario involves the communication unit for sending and receiving messages, the local danger warning application running on a separate ECU or coexisting with other applications on the same ECU. In order to recognize a critical situation, the LDW application is connected to various sensors. Further, a connection to the Human

Machine Interface (HMI) is required for displaying the warning message. Usually, a car sending out a critical warning message does not involve the HMI.

3.6.2 Scenario

Table 5 Scenario: Local Danger Warning to other Cars

Step no.	Actor	Recipient	Type	Data / act	Data length	Remark
1	Sensor/ ECU	–	algo	Tracking of safety critical information		
2	Sensor/ ECU	LDW	com	Sending of information to LDW application	1–8 Byte	
3	LDW	–	algo	Processing of information received from different sensors and applications (e.g. navigation) and recognition of a critical situation.		Application may run on a dedicated ECU or coexist with other application on one ECU.
4	MyCar	Car	com	Cooperative Awareness Message (CAM)	64–256 Byte	
5	CU	LDW	com	Decapsulation of payload within CAM and forwarding to LDW application		in other car
6	LDW	–	algo	Processing of received information		in other car
7	LDW	HMI	com	HMI is triggered and relevant information sent to HMI; information can also be provided to a CSC component.		in other car
8	HMI	–	algo	Warning is displayed		in other car
9	CSC	–	algo	Possible processing of information		in other car

Within this use case, steps 1 to 4 are of major importance. The further steps are already described in Section 3.3.

3.6.3 Requirements

3.6.3.1 Functional requirements

– Reliable communication infrastructure between entities

– Real-time capabilities of the system

– Sensors providing respective data

– Applications that process data and aggregate the information to a local danger warning message.

3.6.3.2 Technical Requirements

Processing time is of major importance in critical situations. Within intersection scenarios a frequency of 10 Hz for sending Cooperative Awareness Messages is required for recognizing and responding appropriately in a critical situation.

In order to prevent a vehicle sending out corrupt data, the on-board network must be secured to prevent attacks on data in steps 1 to 4.

3.6.4 Security Aspects

Within very critical situations, the used data is processed within a very short time frame. It has to be assured that information received from another entity can be checked to be authentic and trustworthy. Only trustworthy data should be processed by the LDW application. Manipulated or unsecured data should be filtered in advance.

3.7 Use Case – (My car 2 Car) Traffic Information to other Entities

3.7.1 Description

3.7.1.1 General

Classic traffic information is limited by high levels of latency and partly by inaccurate information. Enhancing this technology with car2X information from other cars, road-side unit and backend service infrastructure will permit more efficient restructuring of traffic flows.

3.7.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 12).

At least two entities are involved: the car receiving traffic information messages and the entity sending such a message. Usually, for traffic information, the number of entities providing information enhances accuracy of the data.

Within the on-board network, the scenario involves the communication unit for sending and receiving messages, the traffic information application running on a separate ECU or co-existing with other applications on the same ECU. Further, a connection to the Human Machine Interface (HMI) and the navigation system is required on the receiving side (cf. Section 3.4).

For generating a traffic information message, the in-vehicle sensors collect data and send these data to the traffic information application, which processes and aggregates the information. Then, this information is sent out via the communication unit (CU).

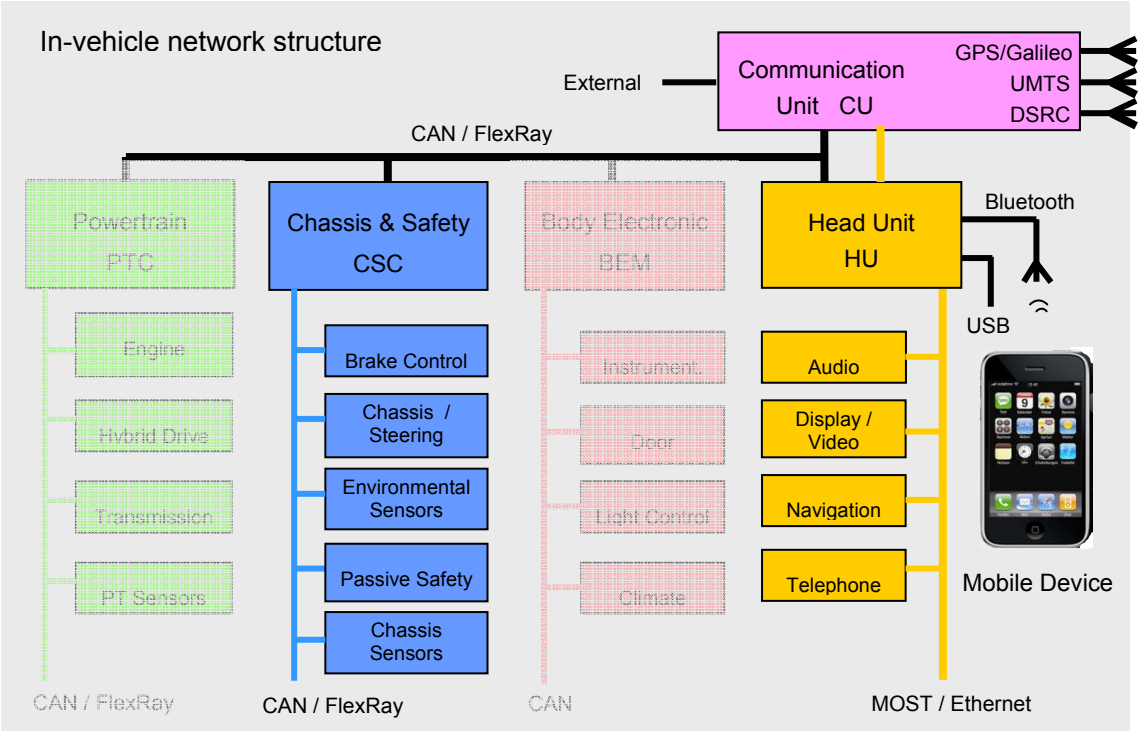

Figure 12 Communication Entities and Relations: Traffic information to other entities

3.7.2 Scenario

Table 6 Scenario: Traffic Information to other Entities

Step no.	Actor	Recipient	Type	Data / act	Data length	Remark
1	Sensors	–	algo	Recognition of relevant traf-fic information via in-vehicle sensors.		
2	Sensor	Traffic Application	com	Sending of data to traffic application		
3	Traffic Application	–	algo	Processing of data and aggregation		
4	Traffic Application	CU	com	Sending of data to TCU		
2	MyCar	Car	com	Traffic Information Message	64–256 Byte	
3	CU	Traffic Application	com	Decapsulation of payload and forwarding to application		in other car
4	Traffic Application	–	algo	Processing of received information		in other car
5	Traffic Application	HMI/ Navigation	com	HMI/Navigation is triggered and relevant information sent to HMI		in other car
6	HMI	–	algo	Warning is displayed		in other car

3.7.3 Requirements

3.7.3.1 Functional Requirements

– Reliable communication infrastructure between entities
– Sensors and application being able to provide traffic information data
– Respective Backend-Infrastructure in order to aggregate traffic information.

3.7.3.2 Technical Requirements

Processing time for traffic information is not as critical as in warning applications.

3.7.4 Security Aspects

For traffic information, the trustworthiness of the data is of major importance. The recipients have to trust the information. Further, the freshness of the data is important in order to have an up to date view on the traffic situation.

It has to be assured as far as possible that the data is not attacked during processing within the on-board system.

3.8 Use Case – (Car2 I and I 2 Car) eTolling

3.8.1 Description

3.8.1.1 General

Car tolling is already in use in different countries using different techniques. Most are based on the same principle: The use of an extra On-Board Unit (OBU). In Germany, for example, the service called “Toll Collect” is used to account trucks. The use case will be described based on the German system. According to the EVITA use cases reference architecture the OBU can be seen as an enhanced CU.

The Toll Collect system [5] provides two types of accounting: the manual accounting and the automatic one. Just the automatic one will be considered within the description of this use case.

To be able to automatically account the trucks, Toll Collect system used the combination of two positioning systems: the Global System for Mobile Communication GSM and the Global Positioning System GPS. Those two technologies are implemented in the Road Side Units (RSU) of the toll provider. In the vehicle, the OBU is equipped with a GPS antenna and GSM antenna in order to communicate with the RSU and to send the relevant information. With the position technologies, the OBU is then able to determine the driven distance in order to calculate the bill based on the driver contract information and in order to send it per mobile phone technology (GSM) to the data processing center of the toll provider.

Considering the fact that for car2X communication a communication unit will be introduced in the car, the logical consequence will be the use of this unit for toll purposes. There-

fore, in the description it is assumed that the OBU as part of the Communication Unit will handle the communication with the RSU.

In this use case, the RSUs of the toll system provider are continually broadcasting a kind of wake-up signal. Depending on its position an OBU recognises a toll road and automatically saves the necessary data for the accounting. Passing the toll provider RSU, the vehicle receives the control signal and the OBU automatically calculates the toll fee. Before sending the needed data for toll accounting the CU checks the origin of the message (authentication of the RSU). If the RSU cannot be identified, the CU does not send any message after the check. Otherwise, it sends the data needed for accounting the driver: the type of the car, toll contract identification, pay method, and the signed bill of the last paid toll.

All the data sent by the CU are signed and encrypted in order to ensure that the driver will be correctly accounted and that only an allowed control center can process the data.

3.8.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 14).

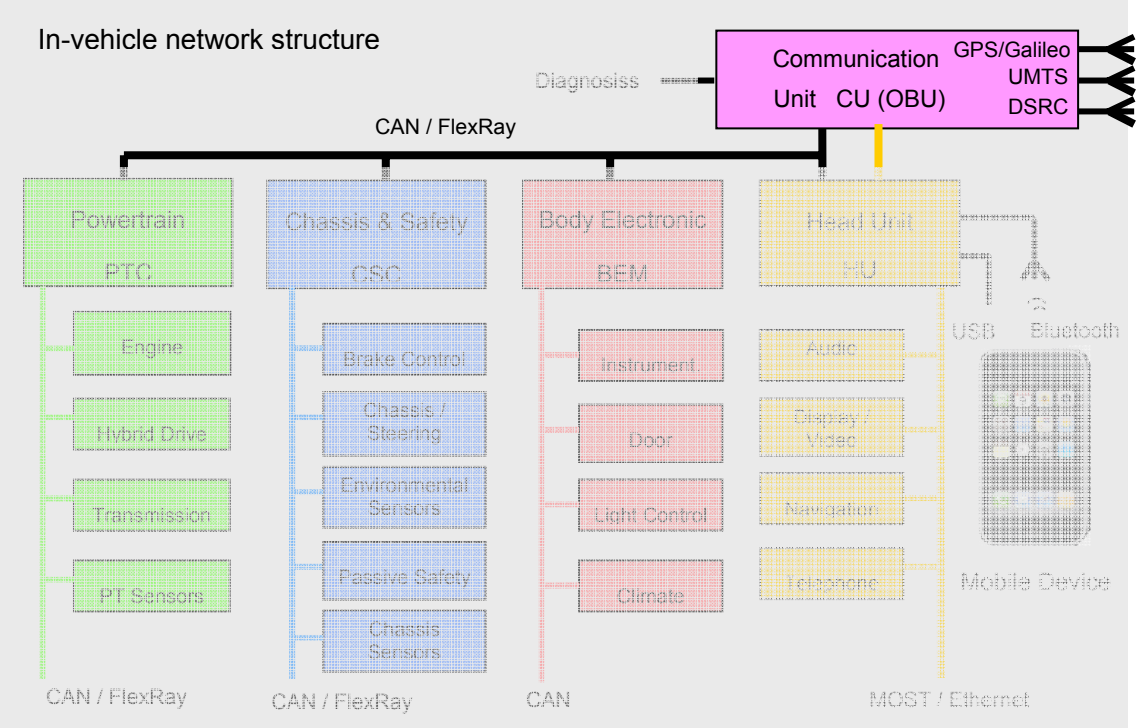

Figure 13 Communication Entities and Relations: eTolling

3.8.2 Scenario

Table 7 Scenario: eTolling

Step no.	Actor	Recipient	Type	Data / act	Data length	Remark
1	OBU (in CU)	–	com	Collecting position data		GPS
2	RSU	OBUs in the area	com	Wake-up signal		
3	OBU (in CU)	–	com	Recognizing toll road		
3	OBU (in CU)	–	algo	Bill calculation		
4	OBU (in CU)	–	algo	Sign and encrypt all relevant info		
5	OBU (in CU)	RSU (Toll center)	com	Sending of info		GSM

3.8.3 Requirements

3.8.3.1 Functional requirements

- OBU with signal reception/sending capabilities through GSM network
- OBU with GPS signal reception and positioning calculation

3.8.3.2 Technical Requirements

- Communication type: Broadcasting, GPS, GSM
- Communication range: GSM Network range
- Maximum delay depends on the communication range

3.8.3.3 Security Aspects

- **Authentication** is an issue since the vehicle must check if the RSU is allowed to receive his billing information.
- **Confidentiality** is necessary because the information is sensitive.
- **Data integrity** is necessary to make sure the account data sent from the car is not manipulated.
- **Non-repudiation** is an issue since the driver should not be able to contest a correctly generated bill.
- **Anonymity** is an issue. For example, the RSU must be able to recognize the vehicle/owner without allowing an eavesdropper who overhears the connection to gain private information about the owner.

3.9 Use Case – (Car2 I and I 2 Car) eCall

3.9.1 Description

3.9.1.1 General

In case of an accident, e.g. detected by the trigger of the airbag, an emergency call is automatically generated. The last positions of the vehicle (position chain) based on GPS/ Galileo signals are also transferred to enable the location of the vehicle. With these measures the delay from the occurrence of the accident to the arrival of the emergency vehicle is minimized.

Today only few vehicles are equipped with this facility and a Service Provider (e.g. OnSTAR) aggregates the position data and then transfers the call to the next PSAP (Public Service Access Point). The vehicle owner has to pay a monthly or annual fee for this service, usually combined with other services. The emergency call is transferred via the Service Provider to the next PSAP. No changes in the infrastructure of the PSAPs are necessary. The use case eCall is based on this approach.

The public European emergency call system currently under development will use the GSM 112 emergency call number (‘112 eCall’) that is automatically linked to the next PSAP today, so a Service Provider is no longer necessary. In addition to the direct communication with the driver, the PSAPs have to be able to deal with the crash data – thus changes to the PSAP infrastructure are necessary.

The European commission has pointed out that with a fully deployed and mandatory eCall system 2500 lives could be saved per year in Europe [6]. The adoption of this approach is therefore being strongly encouraged.

3.9.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 14).

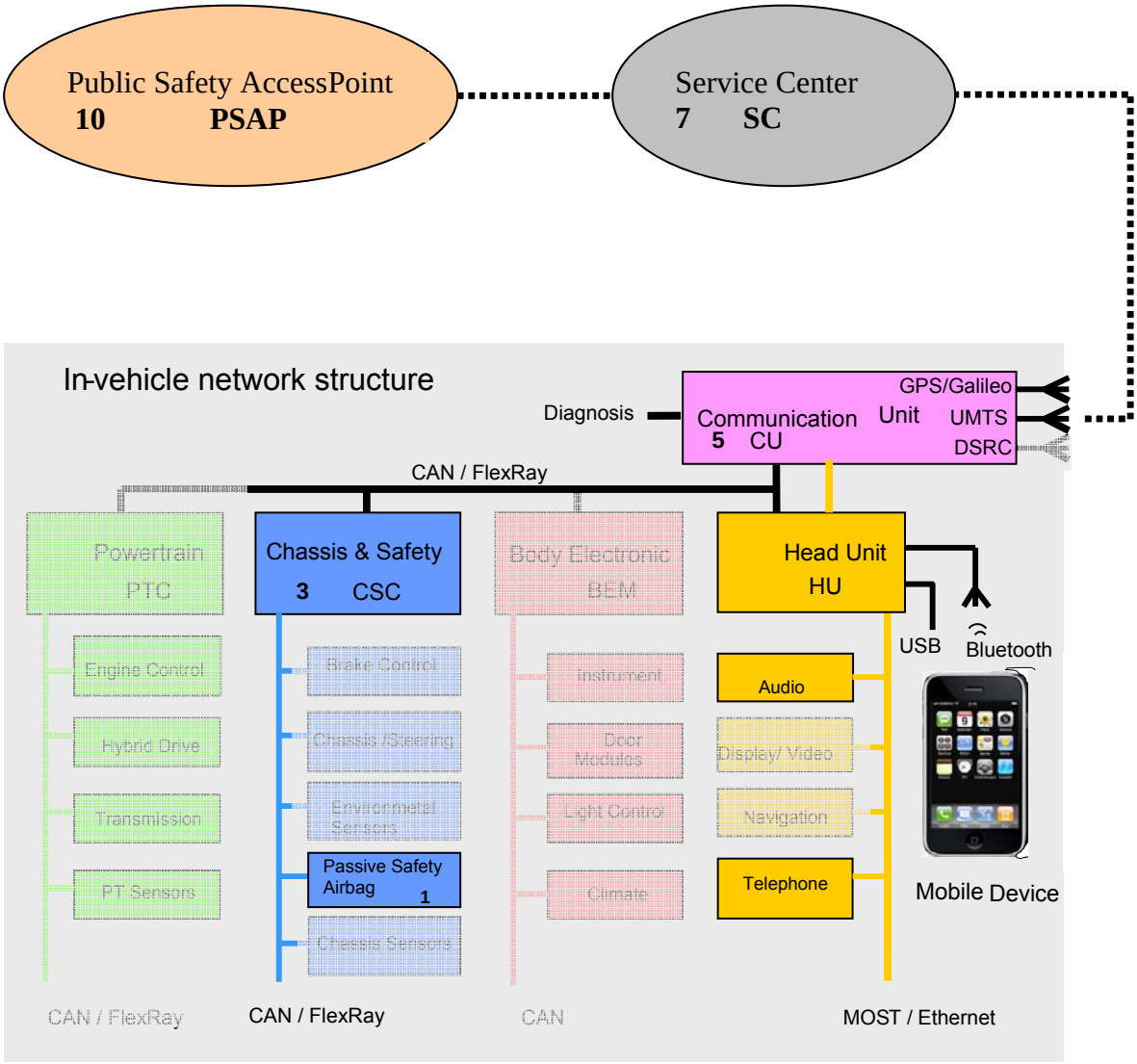

Figure 14 Communication Entities and Relations: eCall

3.9.2 Scenario

Table 8 Scenario: eCall

Step no.	Actor	Recipi-ent	Type	Data / act	Data length	Remark
1	ACU	–	algo	Crash calculation based on internal/external sensors		
2	ACU	CSC	com	Crash info	8 byte	
3	CSC	–	algo	Bus translation		
4	CSC	CU	com	Crash info	8 byte	
5	CU	–	algo	Collect eCall info (Crash & Position Chain)		
6	CU	SC	com	Crash info and position chain	256 byte	
7	SC	–	algo	Contact driver via Telephone		in the service center
8	SC	Driver		Check crash via speech if possible	speech	
9	SC	PSAP	com	eCall / Position Chain	256 byte & speech	in the service center
10	PSAP	–		Sends emergency vehicle		in the service center

3.9.3 Requirements

3.9.3.1 Functional requirements

– Crash Signal has to be secured – normally parallel to airbag deployment (discrete signal line)

– Functionality has to be secured during and after crash (mechanical / electrical)

– PSAP / Service Center have to be able to contact the driver

3.9.3.2 Technical Requirements

– Maximum Delay from the crash to the presence of eCall information in the CU < 1 s. Delay from CU to Service Provider/PSAP depends of the cellular communication infra-structure and the link Service Provider – PSAP.

– The communication from the vehicle has to be maintained until all information is trans-ferred to the Service Provider. Local energy source required.

– Call only to be terminated by the Service Center / PSAP.

3.9.4 Security Aspects

– **Authentication:** False eCalls have to be avoided. This has to be done by authentication in the Service Center.

– **Confidentiality:** The information about the emergency should only be available to the Service Center and the related PSAP.

– **Data integrity** is necessary to make sure the eCall data sent from the car is correct and not manipulated

3.10 Use Case – (Car 2 I and I 2 Car) Remote Car Control

3.10.1 Description

3.10.1.1 General

Enable a remote control of car functions from both outside and inside the vehicle via mobile devices. Possible application examples are closing and opening of windows, doors or similar units with a smart phone. In this use case we describe unlocking and opening of the convertible top from outside of the car with a smart phone.

3.10.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 15).

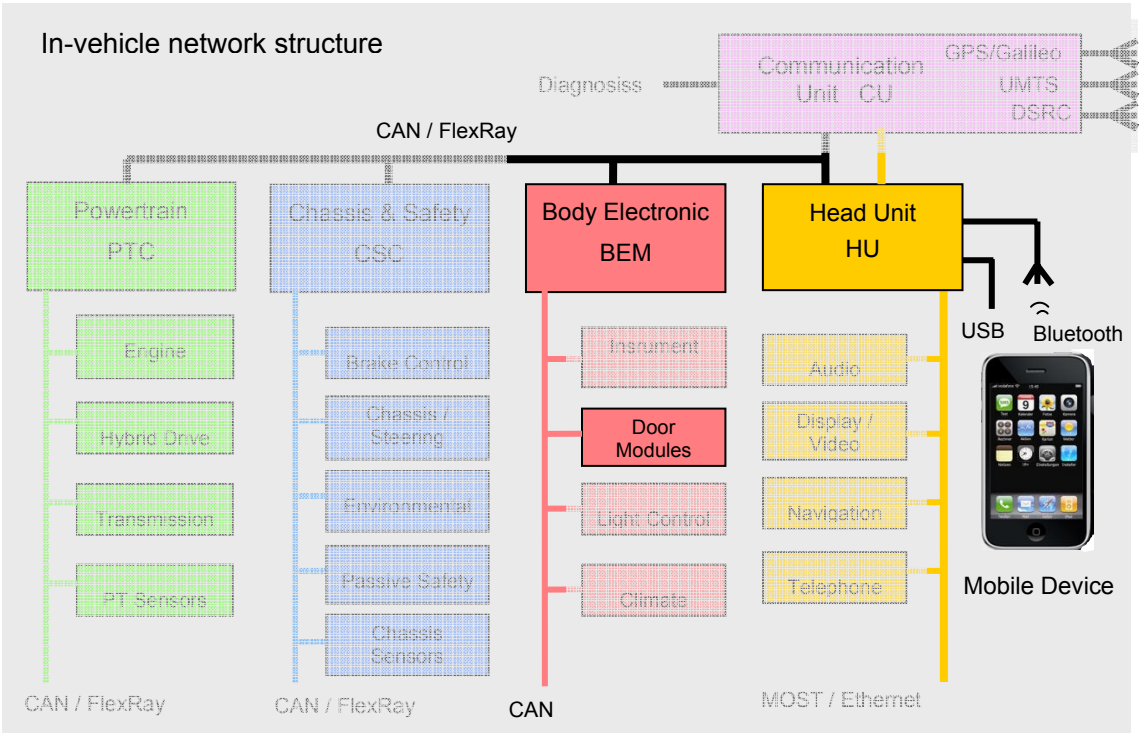

Figure 15 Communication Entities and Relations: Remote Car Control

The use case “remote car control” involves four communication entities, three within the vehicle and one outside of the vehicle. The mobile device outside the vehicle uses a wireless connection, e.g. a smart phone with Bluetooth interface, to connect with the head unit. The head unit is hard-wired to the communication entities within the car.

3.10.1.3 Assumptions, Constraints, Preconditions

It is assumed that a stable and secure software system exists, which when installed on the mobile device will ensure the security and integrity of the connection to the vehicle. In the same way the software also guarantees that the remote control function cannot be activated accidentally.

3.10.2 Scenario

Table 9 Scenario: Remote Car Control

Step no.	Actor	Recipient	Type	Data / act
1	Mobile Device	HU	com	Authentication
2	HU	–	algo	Proof of Authentication
3	HU	Mobile Device	com	Access granted
4	Mobile Device	HU	com	Send Command “Unlock/open hood”
5	HU	BEM	com	Transmit Command
6	BEM	–	algo	Translate Command
7	BEM	Door Modules	com	Transmit Command

3.10.3 Requirements

3.10.3.1 Functional requirements

– Connection interface of the mobile device

– Connection interface in the vehicle head unit

– Hard-wired connection within the vehicle between different entities

3.10.3.2 Technical Requirements

The use case is not time critical, i.e. it is not necessary to minimize the maximum delay into the range of milliseconds, but it should last no longer than 5 seconds. It requires two types of communication:

– wireless communication interface for Bluetooth, FireWire or other standards that are used by the mobile device.

– communication system within the vehicle.

Furthermore it requires that control of the convertible top is implemented within one module, here for example in the door modules.

3.10.4 Security Aspects

Some crucial security aspects are already mentioned before, especially the integrity and non-replicability of the wireless connection between mobile device and car. The systems also have to ensure a highly secure system of authentication, so that it ensures that only registered mobile devices can communicate with the car. Furthermore it should be assured that the mobile device can only be used by authorized persons, e.g. by using identification systems to start the software system for remote car control. Non-repudiation is not necessary as long as there are hard-wired solutions in the car as well.

3.11 Use Case – (Car 2 I and I 2 Car) Point of Interest

3.11.1 Description

3.11.1.1 General

In this use case a service provider offers advertising information through Road Side Units (RSUs). Drivers can receive information about shops, service stations, restaurants, drugstores, etc. Although this kind of information is available through most navigation systems; this could be more suitable for clients, because the information will be up-to-date and not software version dependant as with the navigation software.

We assume in this use case that the RSU just broadcasts the advertising information. We will not consider a distinction of advertising information type. Drivers preconfigure whether they want to receive advertising information.

Entering an area covered by a particular Road Side Unit, the vehicle receives a signed message from the Road Side Unit. The RSU identity is verified. The CU then sends the information to the Head Unit, which then displays it.

3.11.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 16).

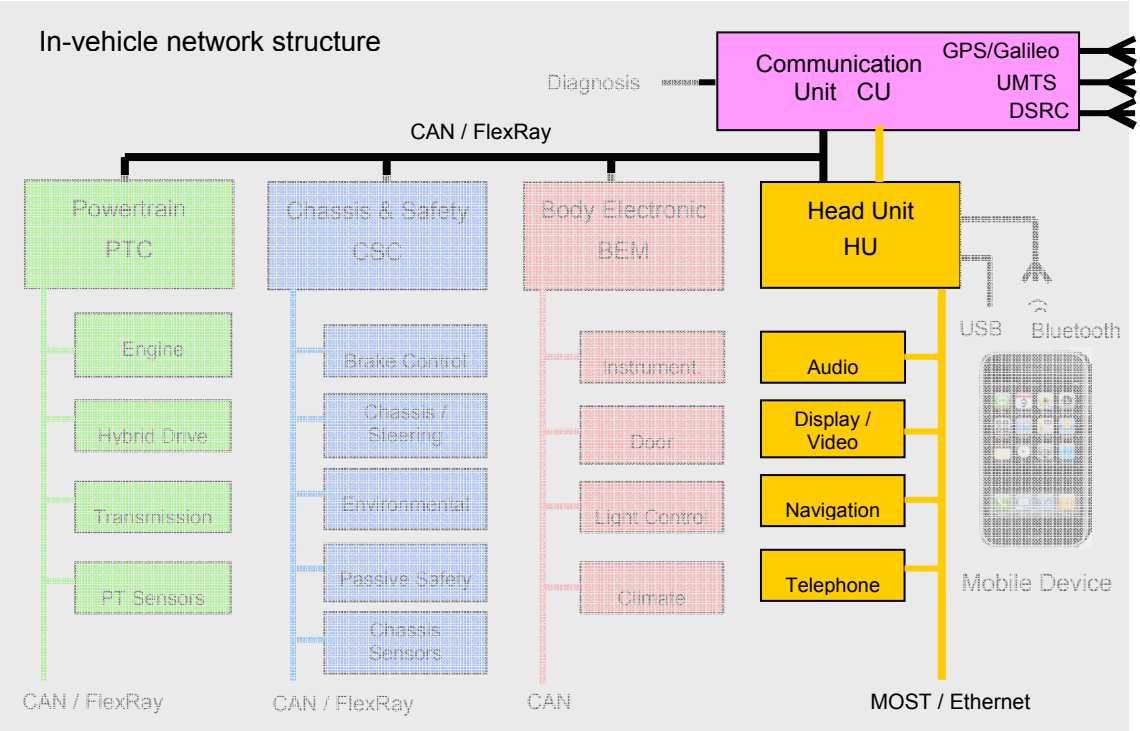

Figure 16 Communication Entities and Relations: Point of Interest

3.11.2 Scenario

Table 10 Scenario: Point of Interest

Step no.	Actor	Recipient	Type	Data / act
1	RSU	CU	com	Signed Message
2	CU	–	algo	Signature verification
3	CU	–	algo	Allowed / Deny
4	CU	HU	com	Message forwarding
5	HU	–	algo	Display

3.11.3 Requirements

3.11.3.1 Functional requirements

– The RSU must be able to broadcast messages

– In-vehicle CU ability to check the identity of a RSU

– In-vehicle CU ability to send allow/deny request

3.11.3.2 Technical Requirements

– Communication type is Broadcast

– Communication range: State-of-the-art

– Maximum delay depends on the communication range

3.11.4 Security Aspects

- **Authentication** is an issue since the recipient should check whether the RSU is allowed to send advertising messages.
- In our case the messages sent from the RSU are not sensitive. They do not need to be encrypted. To the contrary, the intent of the RSU is to publicize the content. Therefore, **confidentiality** is not needed.
- But since the goal is advertising, unauthorized modification of the data should be detected. Therefore, **data integrity** is needed.
- **Non-repudiation** could be used to prove that a particular advertisement was broadcast; e.g. to complain about inappropriate content. This will usually not be necessary, depending on the advertiser's business model.
- **Anonymity** is not an issue. The vehicle does not broadcast any information.

3.12 Use Case – (Nomadic Device) Install applications

3.12.1 Description

3.12.1.1 General

The purpose of this use case is to describe the possibility of installing and running applications in the car from an external device. These can be used through modules by the driver or occupants. Here we exemplify the use case for installing a city application, i.e. visitor guidance for a city, which shows interesting routes and points of interests in the city.

3.12.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 17).

The use case "Install applications" involves six communication entities, four within the vehicle and two outside of the vehicle. The mobile/nomadic device uses a hard-fixed connection, e.g. a USB stick, to connect with the head unit via a USB interface. The head unit is hard-wired connected to the communication entities (Display/Video; Navigation; Communication Unit) within the car. The communication unit has a connection to a positioning system like GPS or Galileo.

3.12.1.3 Assumptions, Constraints, Preconditions

The use case assumes a stable communication connection and interfaces for using mobile devices as well as navigation systems.

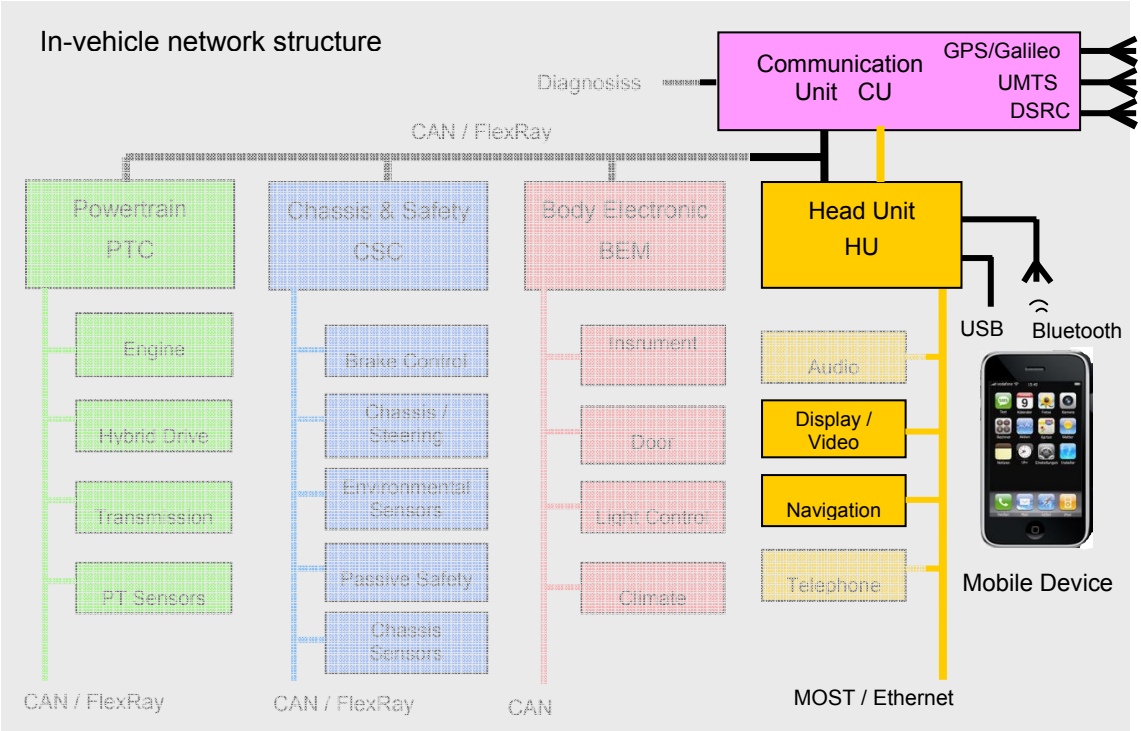

Figure 17 Communication Entities and Relations: Install Applications

3.12.2 Scenario

Table 11 Scenario: Install Applications

Step no.	Actor	Recipient	Type	Data / act
1	Mobile Device	HU	com	Authentication
2	HU	–	algo	Proof of Authentication
3	HU	Mobile Device	com	Access granted
4	Mobile Device	HU	com	Send Data
5	HU	–	algo	Install and run application City Guide
6	HU	Display	com	Send Data
7	Display	User	algo	Run application
8	User	Display	com	Choose destination
9	Display	HU	com	Transmit Destination
10	HU	Navigation	com	Send Destination data
11	Navigation	HU	com	Request own position
12	HU	–	algo	Translate destination data
13	HU	CU	com	Send Request
14	CU	GPS	com	Request Position
15	GPS	CU	com	Transmit Position
16	CU	HU	com	Transmit Position Data
17	HU	–	algo	Translate Position Data
18	HU	Navigation	com	Send Position Data
19	Navigation	–	algo	Calculate way
20	Navigation	Display	com	Send Results
21	Display	–	algo	Show way

3.12.3 Requirements

3.12.3.1 Functional requirements

- Connection interface of the mobile device (here: USB)
- Connection interface in the vehicle head unit (here: USB)
- Hard-fixed connection within the vehicle between different entities
- Connection Interface of the communication unit to positioning systems, e.g. GPS/Galileo

3.12.3.2 Technical Requirements

The use case is not time critical, i.e. it is not necessary to minimize the maximum delay into the range of milliseconds, but it should not last longer than a convenient time span for the user between the connection of the mobile device and the execution of the application. This is also valid for the time span between choosing a destination and displaying results. Due to the data intensive communication within the Head Unit a high-performance, hard-fixed connection is required between them.

3.12.4 Security Aspects

The security and integrity of the installed application has to be ensured. Besides user authentication for the installation process there must be systems of protection like certificates. The data flow between CU and HU concerning position should take place through a secured hard-fixed connection, not through separate connections because of the possibilities to sneak and manipulate data. Furthermore it should be assured that the mobile device can only be used by authorized persons, e.g. by using identification systems to start the software system for application installation. Non-repudiation is not necessary as long as there are hard-fixed solutions in the car as well.

3.13 Use Case – (Nomadic Device) Secure Integration

3.13.1 Description

3.13.1.1 General

The use case demonstrates the integration of an application installed on mobile device, e.g. a media player on a notebook, within the multi media function of the car. This use case demonstrates how a notebook could access the Internet via the connections of the car, download multimedia content, and use the audio and video devices of the car to display these data.

3.13.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 18).

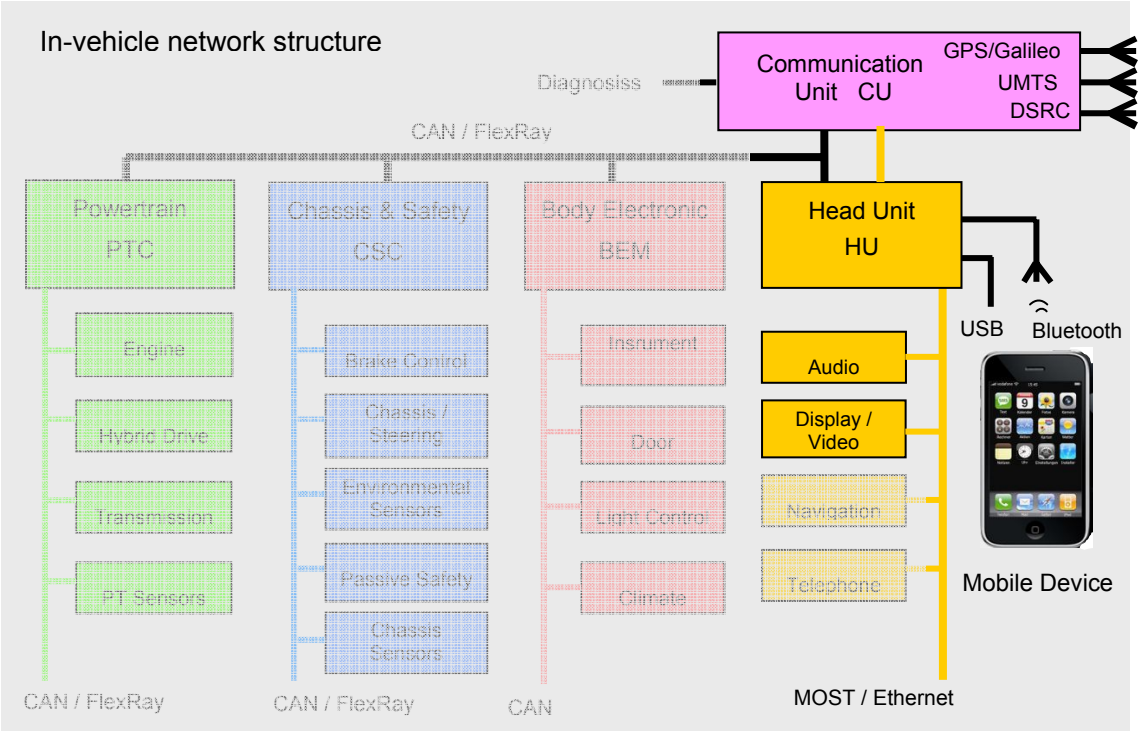

Figure 18 Communication Entities and Relations: Secure Integration

The use case “Secure integration” involves six communication entities, four within the vehicle and two outside of the vehicle. The mobile/nomadic device (Notebook) uses a wireless connection, e.g. Bluetooth, to connect with the head unit. The head unit is hard-wired connected to the communication entities (Display/Video; Audio; Communication Unit) within the car. The communication unit has a connection to the Internet via UMTS.

3.13.1.3 Assumptions, Constraints, Preconditions

The use case assumes stable and secure communication connections and interfaces for the mobile devices as well as for the communication within the car.

3.13.2 Scenario

Table 12 Scenario: Secure Integration

Step no.	Actor	Recipient	Type	Data / act
1	Mobile Device	HU	com	Authentication
2	HU	–	algo	Proof of Authentication
3	HU	Mobile Device	com	Access granted
4	Mobile Device	HU	com	Request Internet Access
5	HU	CU	Com	Send Request
6	CU	–	algo	Process Request
7	CU	UMTS	com	Establish Connection
8	UMTS	CU	com	Transmit Data
9	CU	HU	com	Send Data
10	HU	Mobile Device	com	Receive Data
11	Mobile Device	–	algo	Process Data
12	Mobile Device	HU	com	Request Access Video/Audio device
13	HU	–	algo	Process request
13	HU	Mobile device	com	Allow/Establish connection to A/V
14	Mobile Device	Audio/Video	com	Send Data
15	Audio/Video	–	algo	Process/Display Data

3.13.3 Requirements

3.13.3.1 Functional requirements

– Connection interface of the mobile device (here: Bluetooth)

– Connection interface in the vehicle head unit (here: Bluetooth)

– Hard-fixed connection within the vehicle between different entities

– Connection Interface of the communication unit to Internet, e.g. UMTS

3.13.3.2 Technical Requirements

The use case is not time critical, but it should be convenient for users. The system has to ensure the quality of service for the download as well as for the transmission to the display and sound system. Due to the data intensive communication between the Head Unit and the Communication Unit a high-performance, hard-fixed connection is required between them.

3.13.4 Security Aspects

– It has to be ensured that only registered mobile devices are allowed to communicate with HU and use functionality like internet access.

– It has to be ensured that the internet access by the CU is secured.

– It must be ensured that the mobile device application that uses files downloaded from the internet has to authenticate again before it is allowed to use other devices. This process has to assure that it is not possible that corrupted data (e.g. a virus) can be sent to the HU.
– It should be assured that the mobile device can only be used by authorized persons, e.g. by using identification systems to start the software system for secure integration. Non-repudiation is not necessary as long as there are hard-fixed solutions in the car as well.

3.14 Use Case – (Nomadic Device) Personalize the car

3.14.1 Description

3.14.1.1 General

Enable a driver to personalize a car, i.e. to adjust seat position, mirrors, and preferred settings for multimedia devices, without physical action. Because of this a “User Profile”, which was created once before, will be activated from a mobile device. Here we will exemplify how the seat position becomes adjusted.

3.14.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 19).

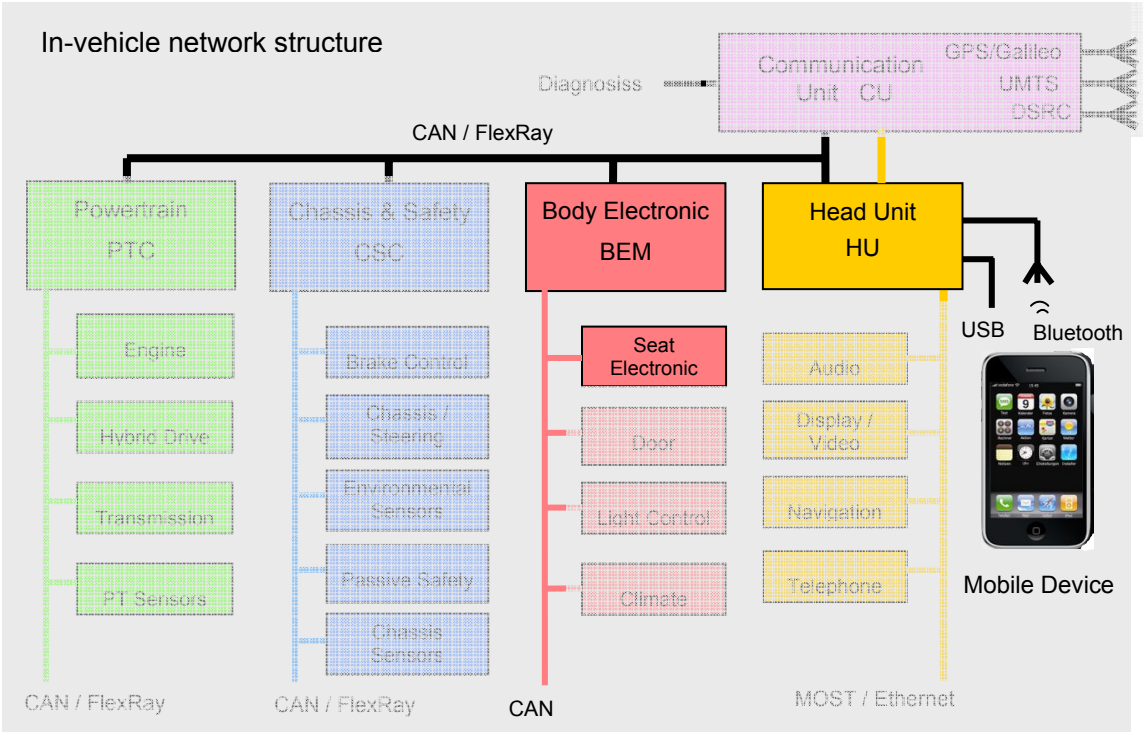

Figure 19 Communication Entities and Relations: Personalize the Car

The use case personalize a car involves four communication entities, three within the vehicle and one outside of the vehicle. The mobile device outside the vehicle uses a wireless connection, e.g. a smart phone with Bluetooth interface, to connect with the head unit. The head unit is hard-wired connected to the communication entities (BEM etc.) within the car.

3.14.1.3 Assumptions, Constraints, Preconditions

The use case assumes that a stable and secure software system exists, which when installed on the mobile device will ensure the security and integrity of the connection to the vehicle. It is assumed that the data of the user profiles is stored in the Head Unit.

3.14.2 Scenario

Table 13 Scenario: Personalize the Car

Step no.	Actor	Recipient	Type	Data / act
1	Mobile Device	HU	com	Authentication
2	HU	–	algo	Proof of Authentication
3	HU	Mobile Device	com	Access granted
4	Mobile Device	HU	com	Send Request for User profile X
5	HU	–	algo	Read User Profile
6	HU	BEM	com	Transmit Command Seat Position
7	BEM	–	algo	Translate Command Seat Position
8	BEM	Seat Electronic	com	Transmit Command
9	Seat Electronic	–	algo	Change Seat Position

3.14.3 Requirements

3.14.3.1 Functional requirements

– Connection interface of the mobile device, here: Bluetooth

– Connection interface in the vehicle head unit, here: Bluetooth

– Hard-fixed connection within the vehicle between different entities

3.14.3.2 Technical Requirements

The use case is not time critical, i.e. it is not necessary to minimize the maximum delay into the range of milliseconds, but it should be convenient for the user. It requires two types of communication:

– wireless communication interface for Bluetooth, FireWire or other standards that are used by the mobile device.

– communication system within the on-board network.

3.14.4 Security Aspects

It has to be ensured that the integrity and non-replicability of the wireless connection between the mobile device and the car is achieved. The systems also have to ensure a highly secure system of authentication, so that it ensures that only registered mobile devices can communicate with the car. Furthermore it should be assured that the mobile device can only be used by authorized persons related to the stored profile, e.g. by using identification systems to start the software system on the smart phone.

3.15 Use Case – (Aftermarket) Replacement of Engine ECU

3.15.1 Description

3.15.1.1 General

Due to a malfunction the engine control ECU of a vehicle has to be replaced. Normally this is done by an authorized garage, when the diagnosis shows that the reason for the malfunction is the ECU. If the ECU is capable of being flashed, then the garage has to install exactly the right version of the hardware of the ECU and then download the right software version as described in the “Remote Flashing” and “Flashing per OBD” use cases. If the software cannot be downloaded, e.g. for older cars with ROM (Read-Only Memory), the garage has to install the right ECU hardware with the correct software version.

Remark: The processing of the software download and the related communication scenarios are described in the “Remote Flashing” and “Flashing per OBD” use cases.

3.15.1.2 Communication Entities and Relations

Figure 20 shows the function split and communication interfaces in normal driving state after replacement of the engine ECU and software download.

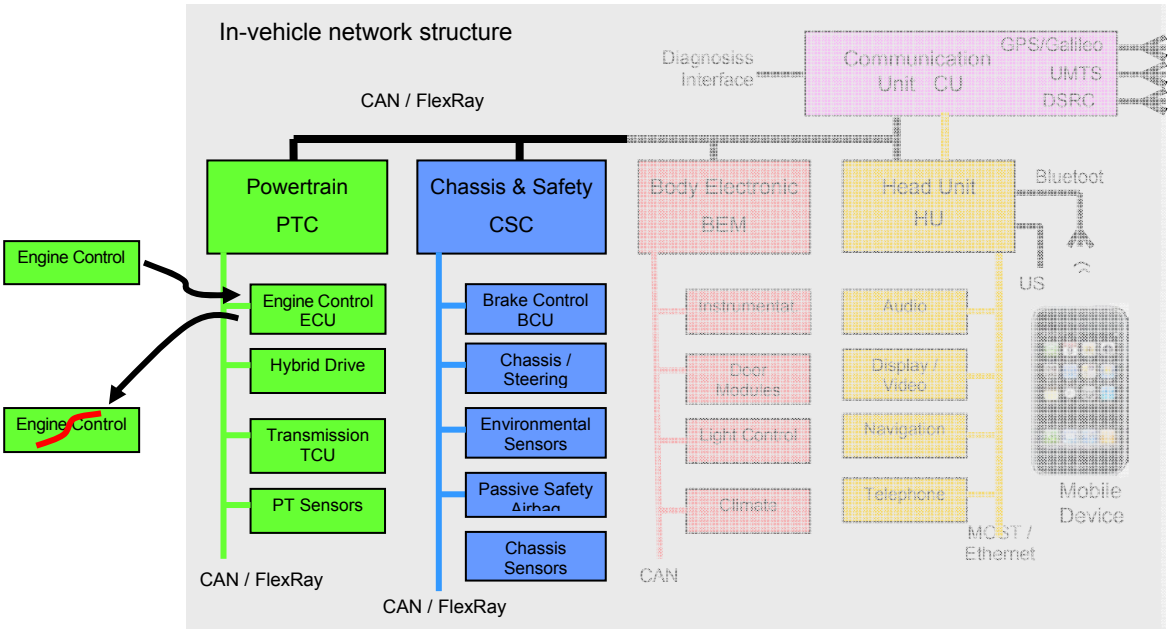

Figure 20 Communication Entities and Relations: Replacement of Engine ECU

3.15.2 Scenario

Table 14 Scenario: Replacement of Engine ECU

Step no.	Actor	Recipient	Type	Data / act	Data length	Remark
1	ECU	–	algo	Calculation and control of requested engine torque according to input of the driver and via communication		Typical cycle of engine control: 10 ms
2	ECU	TCU, PTC	com	Engine torque (amongst others)	8 Byte (CAN message)	Every 10 ms
3	PTC	CSC	com	Transfer ECU data	ditto	
4	CSC	BCU	com	Transfer ECU data	ditto	
5	BCU	–	algo	Calculates wheel-individual slips due to wheel-speed-sensor signals. Applies brake force to the driven wheels if necessary.		Typical cycle wheel speed sensing and brake control: 10 ms
6	BCU	CSC	com	Engine torque command to ECU due to wheel slips. If driven wheels are spinning engine torque has to be reduced	8 Byte (CAN message)	
7	CSC	PTC	com	Transfer BCU data	ditto	
8	PTC	ECU	com	Transfer BCU data	ditto	Typical cycle Step1-8: 50 ms
9	ECU	–	algo	engine torque control due to driver and BCU input (amongst others) → Step 1		

3.15.3 Requirements

3.15.3.1 Functional requirements

- The new ECU must have the same function and communication scheme as the replaced ECU.
- The new ECU must have a release of the OEM.
- The new ECU must perform a self-check of the Engine System to verify correct system configuration.
- In case of software flashing all functional requirements described in use case “Flashing per OBD”.

3.15.3.2 Technical requirements

– Due to the fact that safety functions are based on the communication, it is necessary to verify the integrity of the transferred data via checksum, etc.

– In case of software flashing all technical requirements described in use case “Flashing per OBD”.

3.15.4 Security Aspects

Also in driving state (not only in flashing mode) the integrity of the transferred data is required for safety reasons.

In case of software flashing in the garage security aspects are described in use case “Flashing per OBD”.

3.16 Use Case – (Aftermarket) Installation Car2x Unit

3.16.1 Description

3.16.1.1 General

In a garage a Car2X Communication Unit (CU) is installed into a car. The car was not equipped with a CU before. The installation will include mechanical installation, connection to power supply, connection to the backbone bus of the car and mounting of antennas and antenna cabling. All configuration work that is needed to associate the CU with the car, to allow payment function etc., is done in the garage.

The functionality of the car will be only changed in that new information from the CU can be accepted and integrated in the procedures of the installed car systems.

The information that is broadcast from the CU will be taken from the backbone bus. To accept and use the data from the CU, the Chassis Safety Controller (CSC) and the Head Unit (HU) will need software updates that have to be installed during the installation.

Remark: The processing of messages from and to the CU is described in detail in use case “Safety reaction: Active brake” and use case “Message leads to safety reaction”.

3.16.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 21).

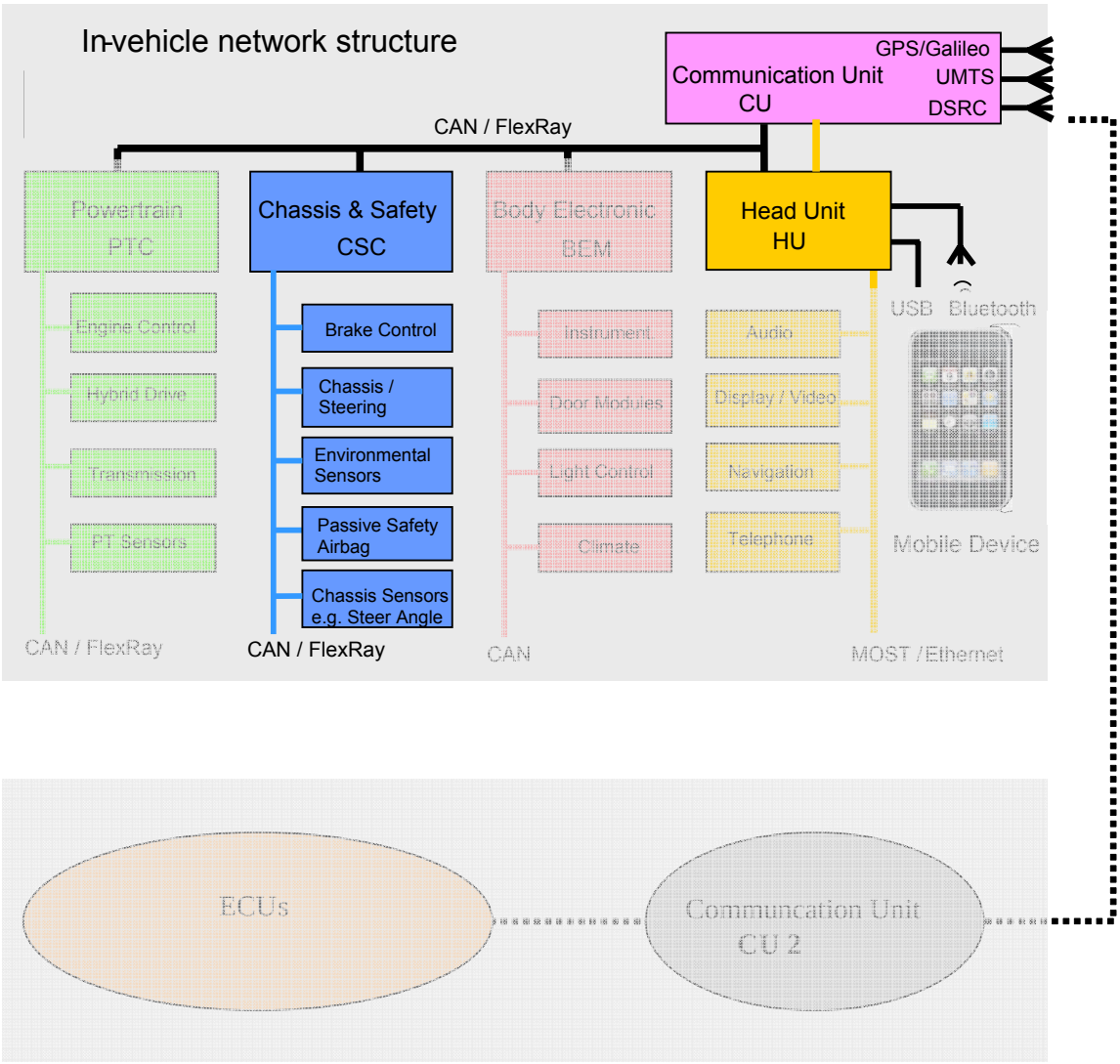

Figure 21 Communication Entities and Relations: Installation Car2X Unit

3.16.2 Scenario

Communication scenarios of the installed car2X CU are described in use case “Safety reaction: Active brake” and use case “Message leads to safety reaction”.

3.16.3 Requirements

3.16.3.1 Functional requirements

– The new CU should be a communication node with the same level of confidence as a CU that is installed during production.

– The new CU should be a communication node with the same functions as a CU that is installed during production.

- The new CU should allow the same functions in the CSC and HU as a CU that is installed during production.
- The safety level of all car ECUs should be unchanged.
- The safety level of all car busses should be unchanged.
- No function of the car may be degraded by the new Car2X functions.
- The new CU can present itself only as a car CU to other communication partners. It can not (by chance, failure or intention) present itself as a privileged CU (Privileged CUs are reserved for emergency vehicle, police cars etc.).
- Privacy of the broadcast car information has to be guaranteed.
- If the CU is stolen it cannot be used to simulate a false identity of a car.
- If the CU is stolen it cannot be used to simulate a false identity of a person.
- If the CU is stolen it can not be used to gain access to the financial resources of the legal owner.
- No permanent connection between the CU and a central system should be needed.
- It must be possible to shut down the car and the CU for prolonged periods of time (e.g. several months) without loss of functionality.

3.16.3.2 Technical Requirements

- Additional security information on the busses in the chassis and powertrain domain should be less than 15% of the net data.

3.17 Use Case – (Diagnosis) Remote Diagnosis

3.17.1 Description

3.17.1.1 General

Diagnosis of cars is not a new goal in the automotive industry. It has existed since cars were first designed. During the last 20 years car diagnosis gained more importance because of the increasing use of electronics in cars. Standards were defined not just to allow different manufacturer's ECUs to communicate with each other in an in-vehicle network system, but also to allow different diagnosis tools to have access to diagnosis data; e.g., failure log entries. Those entries are composed of failure codes, their states, and the context in which the failures occurred.

We distinguish between two different types of diagnosis: On-Board Diagnosis and Off-Board Diagnosis. The difference is that an Off-Board diagnosis is done with an off-vehicle system (e.g. diagnosis tool). It is important to mention that an Off-Board Diagnosis can be done by connecting the diagnosis tool to the On-Board Diagnosis system.

For better understanding, a few words about failure log entries. Each ECU has a diagnosis routine, which records failure events (e.g. sensor failure) in the failure log. Since the failure events can be sensitive for different ECUs, different failure records are made. A diagnosis tool will try to know where the real cause comes from, based on two points: the different entries made in different timeframes and the algorithms implemented.

Nowadays car diagnosis in Europe is hardwired. A wireless car diagnosis will be described in our case with focus on the communication characteristics of the data transmission.

In this use case, a car owner wants his car to be inspected by a service station. After receiving the request of the car owner, a service station using a diagnosis tool will try to assess the state of a vehicle located in their area without making any physical connection to the vehicle. The diagnosis of the vehicle should even be possible if the vehicle is not in the area of the service station, by using an internet connection. This is necessary since real time data when a vehicle is moving can help to discover malfunctions, which are not detectable when the car is in the service area.

The service station has to first connect via Internet and Wireless LAN to the in-vehicle network. An employee of the station using the diagnosis tool sends a connection request to the vehicle. The authorization for the connection is checked in the Communication Unit (CU). The message is checked for integrity and the service station is authenticated. A connection answer is sent back to the diagnosis tool. Once the connection is established, the diagnosis tool sends, depending on the option chosen by the employee of the service station, requests to read out diagnosis information (State/Log information) from the Electronic Control Unit (ECU) it wants to check. A motor diagnosis will be considered in this case; therefore, information from the engine control unit shall be read. After receiving the connection request, the CU forwards it to the ECU. A secure session is negotiated between the ECU and the diagnosis tool using a challenge response process.

3.17.1.2 Communication Entities and Relations

The function split and communication interfaces for remote diagnosis are shown in the reference architecture (see Figure 22).

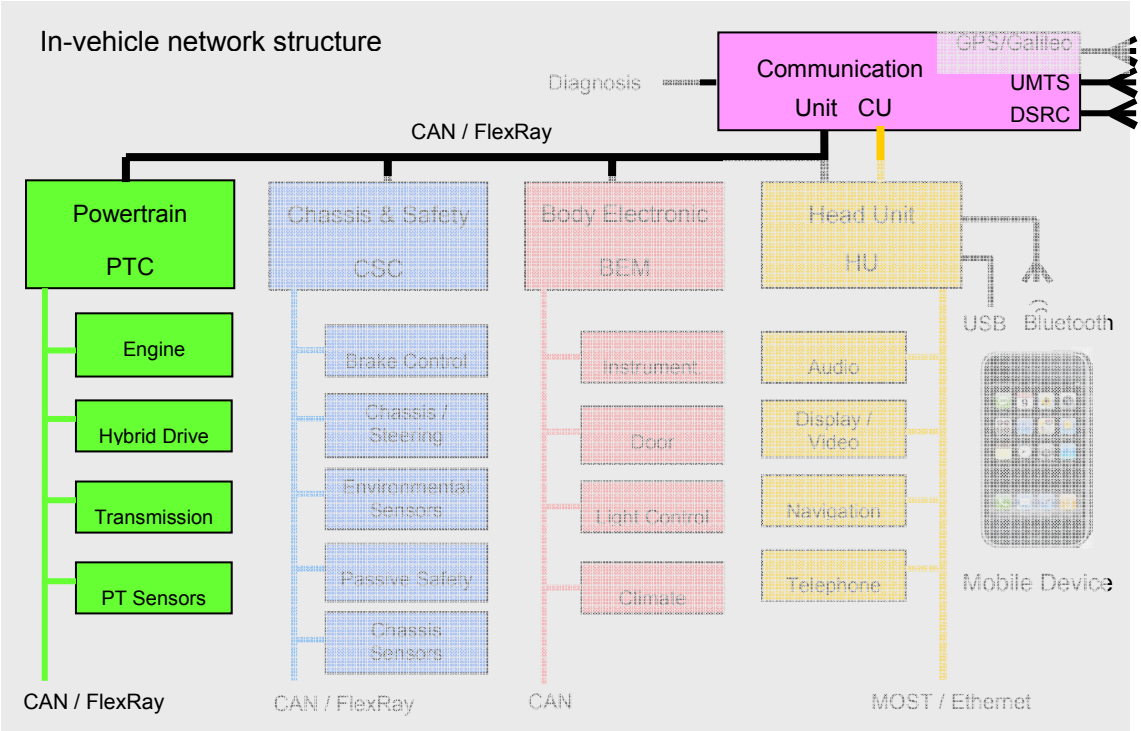

Figure 22 Communication Entities and Relations: Remote Diagnosis

3.17.2 Scenario

Table 15 Scenario: Remote Diagnosis

Step no.	Actor	Recipient	Type	Data / act	Remark
1	Diag.->CU	PTC->ECU	com	Connection request	CAN, FlexRay
2	PTC->ECU	–	algo	Request processing / authentication, integrity check	
3	PTC->ECU	CU->Diag.	com	Challenge	CAN, FlexRay
4	Diag.->CU	PTC->ECU	com	Response	CAN, FlexRay
5	PTC->ECU	CU->Diag.	com	Connection response	CAN, FlexRay
6	Diag.->CU	PTC->ECU	com	Memory Info (state, log) request	CAN, FlexRay
7	ECU	–	algo	Authentication / Integrity check	
8	ECU->PTC	CU->Diag.	com	Memory Info response	CAN, FlexRay

3.17.3 Requirements

3.17.3.1 Functional requirements

- Diagnosis Tool with remote functionality
- In-vehicle CU ability to check the identity of a RSU/ Diagnosis Tool
- In-vehicle CU ability to send/receive external requests
- In-vehicle CU ability to check the data integrity
- Reliable Transmission
- Secure Transmission

3.17.3.2 Technical Requirements

- Communication type: Internet, Wireless LAN
- Communication Range: State-of-the-art
- Maximum delay depends on the communication range

3.17.4 Security Aspects

- **Authentication** is an issue since the vehicle must check if the RSU/Diagnosis Tool is allowed to communicate with it.
- **Data Authenticity (Integrity)** is necessary, since a wrong diagnosis can lead to incorrect actions such as replacing a functional ECU.
- **Data Freshness** to ensure old data is not replayed.
- **Non-Repudiation** can be necessary for example to avoid the repudiation of a wrongdoing by the service station.
- **Anonymity** is an issue. For example, the service station must be able to recognize the vehicle / owner without allowing an eavesdropper who overhears the connection to gain private information about the owner.

3.18 Use Case – (Diagnosis) Remote Flashing

3.18.1 Description

3.18.1.1 General

In the use case “Remote Diagnosis”, the process to connect remotely from a service station to the in-vehicle system of a car has been described. Diagnosis is used to assess the state of the vehicle. A possible consequence of diagnosis would be the update of the software version of the Electronic Control Unit ECU to remove bugs or to improve the functionality. Nowadays this is done over cables, flashing per OBD.

In this use case flashing an ECU wirelessly will be addressed. On one hand this brings advantages such as faster updates, comfort and money savings for the driver, and more clients per day served for the service station since the driver does not have to use the area of the service station to let his car be repaired. On the other hand, this brings a lot of security issues.

We assume that the driver and the service station have an arrangement about the remote flashing of the driver's vehicle.

The service station using a Diagnosis/Flashing Tool establishes a connection via Internet and Wireless LAN to the in-vehicle network. It sends a connection request to the ECU in the Powertrain domain via the Communication Unit (CU). The request is checked for integrity and the service station is authenticated. A connection answer is then sent and session keys are shared to allow a secure communication channel. To know which version will be installed, a diagnosis of the vehicle is done to have all necessary information such as ECU type, Firmware Version, and date of the last update. If the type is the expected one, then the flashing session is started. The flashing tool sends a request to open a programming session at the ECU level.

Once the programming session is open, the flashing tool sends the encrypted new software version to the RAM of the ECU. The communication still goes through the CU. Every message is checked for integrity, authenticity, and freshness at the ECU level. The software is flashed in the ROM, and the date is saved. At the end the flashing tool closes the programming session at the ECU level and the connection with the vehicle.

3.18.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 23).

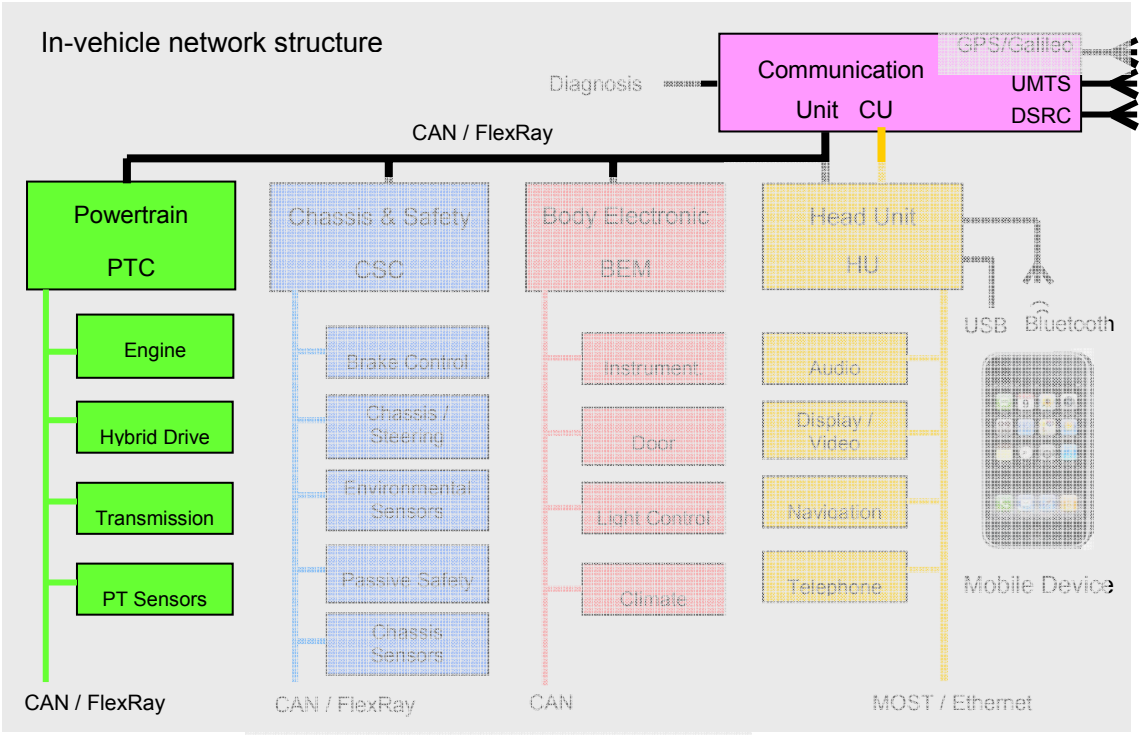

Figure 23 Communication Entities and Relations: Remote Flashing

3.18.2 Scenario

Table 16 Scenario: Remote Flashing

Step no.	Actor	Recipient	Type	Data / act	Remark
1	Service Station->CU	PTC-> ECU	com	Connection request	CAN
2	ECU	–	algo	Request processing / Integrity, Authentication check	
3	ECU-> PTC	CU->Service Station	com	Connection response	CAN
4	Service Station->CU	PTC-> ECU	com	Info request about ECU	CAN
5	ECU	–		Authentication / Integrity check, freshness	
6	ECU-> PTC	CU->Service Station	com	Info response about ECU	CAN
9	Service Station->CU	PTC-> ECU	com	Encrypted Firmware update	CAN
10	ECU	–	algo	Authentication / Integrity check, freshness	
11	ECU	–	algo	Encrypted Firmware update	
12	PTC	–	algo	Decryption and Update execution	

3.18.3 Requirements

3.18.3.1 Functional requirements

- Diagnosis Tool with remote functionality
- In-vehicle CU ability to check the identity of a RSU/ Diagnosis Tool
- In-vehicle CU ability to check the data integrity and data freshness
- In-vehicle CU ability to send/receive external requests
- Reliable connection

3.18.3.2 Technical Requirements

- Communication type: Internet, Wireless LAN
- Communication Range: State-of-the-art
- Maximum delay depends on the communication range#
- Enough memory per ECU for the storage of keys: 48 bytes if an AES-128 bit and an ECC-256 bit is considered.

3.18.4 Security Aspects

- **Authentication** is an issue since the vehicle must check if the RSU / Diagnosis Tool are allowed to communicate with it.
- **Data Authenticity (Integrity)** is necessary, since a wrong diagnosis can lead to incorrect actions such as replacing a functional ECU. Furthermore, it ensures the intent update is flashed.
- **Data Freshness** to ensure that old data is not replayed.
- **Non-Repudiation** can be necessary for example to avoid the repudiation of a wrongdoing by the service station.
- **Confidentiality** is necessary for copy protection: preventing re-engineering, protecting know-how.
- **Anonymity** is an issue. For example, the service station must be able to recognize the vehicle / owner without allowing an eavesdropper who overhears the connection to gain private information about the owner.

3.19 Use Case – (Diagnosis) Flashing per OBD

3.19.1 Description

3.19.1.1 General

In use cases “Remote flashing” and “Remote Diagnosis”, the connection for diagnosis purpose is done wirelessly. Nowadays in Europe, car diagnosis is done hardwired. It’s interesting to take a closer look at the use case, to identify the security issues service stations and vehicles owner are already confronted with. The description is based on the Standard Unified Diagnostics Services UDS, which is specified in [7]. In this use case an ECU firmware of a vehicle will be updated hardwired from a service station.

A car owner takes his car to the area of a service station. To start the diagnosis session the car has to be activated. The ECU initializes its software and starts the diagnosis function, called diagnosis server. In this state, the diagnosis server is in the default mode (this is defined as a session in [7]).

The service station employee connects his diagnosis tool to the on-board diagnosis interface in the vehicle. This is done by plugging a cable to the diagnosis connector, which is different from car to car.

A diagnosis request is then sent via the Communication Unit CU (on-board diagnosis interface) to the ECU. The ECU authenticates the diagnosis tool and checks the data integrity. If the request is successful, the ECU opens a programming session.

The service station employee begins his diagnosis by checking the ECU type and firmware version. Assuming the ECU type is known, a comparison is also made to figure out the need of an update of the version. The diagnosis tool then sends the encrypted packets of the new firmware to the ECU, which stores it in the RAM.

The new firmware is decrypted at ECU level and flashed in the ROM packet wise. The date of the update is written in the ECU and the programming session is closed by sending an EcuReset request to the ECU.

3.19.1.2 Communication Entities and Relations

The function split and communication interfaces are shown in the reference architecture (see Figure 24).

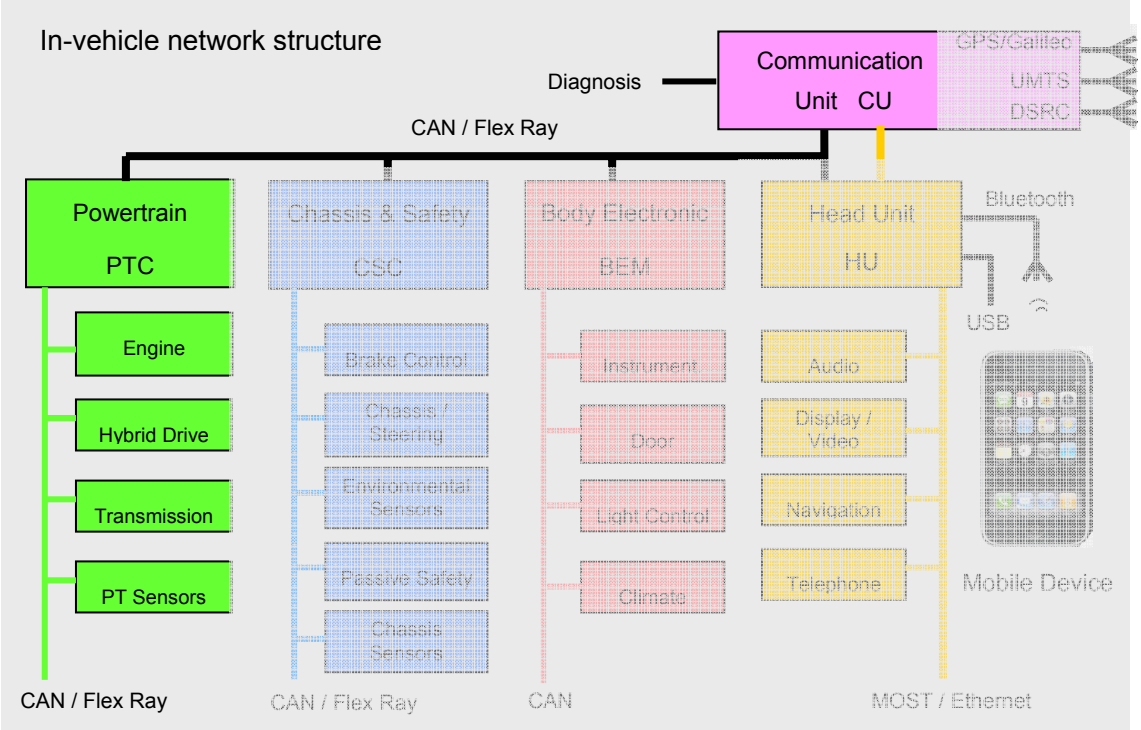

Figure 24 Communication Entities and Relations: Flashing per OBD

3.19.2 Scenario

Table 17 Scenario: Flashing per OBD

Step no.	Actor	Recipient	Type	Data / act
1	Diagnosis Tool	CU	Com	Connection request by plugging the cable
2	CU	PTC -> ECU	Can	Connection request forwarded
3	ECU	—	Algo	Request processing /Authentication check
4	ECU	PTC -> CU	Can	Connection response forwarded
5	CU	Diagnosis	Com	Connection response forwarded
6	Diagnosis Tool	CU	Com	Info (type, version) request about ECU
7	CU	PTC -> ECU	Can	Info request about ECU forwarded
8	ECU	PTC -> CU	Can	Info response about ECU
9	CU	Diagnosis Tool	Com	Info response about ECU
10	Diagnosis Tool	CU	Com	Encrypted Firmware update packet
11	CU	PTC -> ECU	Can	Encrypted Firmware update packet
12	ECU	–	Algo	Integrity check / Decryption of the firmware update /Execution
13	ECU	PTC -> CU	Can	Ok response
14	CU	Diagnosis Tool	Com	Ok response
15	Diagnosis Tool	CU	Com	Close session
16	CU	PTC -> ECU	Can	Close session
17	ECU	–		Session closed

3.19.3 Requirements

3.19.3.1 Functional requirements

- In-vehicle CU ability to check the identity of a RSU/ Diagnosis Tool
- In-vehicle CU ability to check the data integrity
- In-vehicle CU ability to send/receive external requests
- Reliability of the connection

3.19.3.2 Technical Requirements

- Communication type: flash cable connection
- Communication Range unknown: Length of the cable
- OBD Plug in the vehicle

3.19.4 Security Aspects

- **Authentication** is required to avoid the compromising of the in-vehicle system by malicious code.
- **Confidentiality** is required to protect the know-how in the ECU update software.
- **Data Integrity** is necessary to make sure the update software was not manipulated before being executed.
- **Freshness** is required for the messages to be protected against replay attacks.
- **Non-Repudiation** is needed for example to avoid the repudiation of a wrongdoing by the service station.

4 Summary and Outlook

4.1 Summary

The use cases described in this report cover a broad range of possible applications that are either available today or will become available in the future as car2X functionality is deployed in the market. These use cases are representatives of various different alternatives that may be possible in the real world.

Main characteristics of the on-board networks in terms of the subsequent security requirements analyses are:

– Today, usually no specific security measures are used for in-vehicle communication. Safety systems usually use plausibility checks of different signals to detect failures.

– Security measures are used only if the communication is not restricted to the in-vehicle network (e.g. diagnosis, software flashing, e-Tolling).

– The in-vehicle network, especially for the powertrain and chassis & safety domains is designed for “hard” real-time requirements. It has limited bandwidth, but very short response time.

4.2 Outlook

External communication interfaces, fixed and wireless, increasingly become an integral part of automotive on-board architectures. This development is not the least driven by future safety application scenarios. E-safety applications based on car2X communication have been identified as a measure for decreasing the number of fatal traffic accidents. Examples for such e-safety applications are local danger warnings, traffic light pre-emption, or traffic information via road-side units [8]. While these functionalities herald a new era of safety in transportation, new security requirements need to be considered in order to prevent attacks on these systems. Attacks can be manifold: illegally forced malfunctioning of safety critical in-vehicular components as well as the illegal influence of traffic provoked by means of fake messages [9] are just two likely possibilities. With respect to the corresponding risks, the following security considerations are important:

– The in-vehicular system infrastructure must not allow to be illegally tampered with, ensuring that internal safety critical systems cannot be influenced, and the vehicle cannot be provoked to send fake information. Respective attacks on the system have to be prevented, or at least detected and contained.

– Attacks on external communication infrastructure must be prevented, or at least detected and contained, so that the privacy of the communicating entities is preserved and fake messages injected into the (wireless) communication infrastructure would be properly identified and eliminated before influencing e-safety applications.

The EVITA deliverable D2.3 “Security requirements for automotive on-board networks based on dark-side scenarios” deals with the derivation of security requirements for automotive on-board networks from the present use case descriptions.

References

[1] Car 2 Car Communication Consortium Manifesto. Overview of the C2C-CC System. Version 1.1, August 2007, www.car-2-car.org/fileadmin/downloads/C2C-CC_manifesto_v1.1.pdf

[2] Secure Vehicle Communication, European FP6 project SEVECOM, www.sevecom.org

[3] EASIS (Electronic Architecture and System Engineering for Integrated Safety Systems) Deliverable D0.2.4: General Architecture Framework. Version 1.1, 2004, www.easis.org

[4] www.zercustoms.com/news/BMW-Future-Safety-Systems.html (26 Oct 2007)

[5] http://www.toll-collect.de

[6] www.esafetysupport.org/download/European_Commission/COM(2007)%20541intelligent%20car.pdf

[7] International Standard ISO 14229-1:2006. *Road vehicles – Unified diagnostic services (UDS) – Part 1: Specification and requirements*

[8] Kosch, T.: Local Danger Warning based on Vehicle Ad-hoc Networks: Prototype and Simulation. In *Proceedings of 1st International Workshop on Intelligent Transportation* (WIT), Hamburg, Germany, March 2004.

[9] Barisani, A.; Bianco, D.: Unusual Car Navigation Tricks. In *Proceedings of CanSecWest*, Vancouver, Canada, April 2007.

[10] Steffen, R.; Bogenberger, R.; Hillebrand, J.; Hintermaier, W.; Winckler, A., Rahmani, M.: Design and Realization of an IP-based In-car Network Architecture. In *First Annual International Symposium on Vehicular Computing Systems*, Trinity College Dublin, Ireland, July 22-24, 2008