Master report, IDE 1263, June 2012
Embedded and Intelligent Systems

On-Board Diagnostics over Ethernet

Master thesis
School of Information Science, Computer and Electrical Engineering

Saeed Moradpour Chahaki

On-Board Diagnostics over Ethernet

Master Thesis in Embedded and Intelligent Systems

School of Information Science, Computer and Electrical Engineering

Halmstad University

Box 823, S-301 18 Halmstad, Sweden

June 2012

Preface

The current work has been done during the Spring semester 2012 at Scania CV AB, in “Diagnostic Architecture and Product Data” department as a Master thesis. The report aims at investigating the current works and protocols in this domain, addressing possible risks and proposing models and a solution for implementation that is consistent with manufacture requirements.

I highly appreciate my wife’s patience and her emotional support along the way of this degree work and I am proud to have had the supervision of Prof. M. Jonsson from Halmstad University and Mr. P. Sundbäck from RESD department at Scania AB CV. Many thanks to those helped me from REIV department, also. Special thanks to Mr. M. Brolin and Mr. J. Stenarson and Mr. M. Gyllenros with their technical advice.

Saeed Moradpour Chahaki

Halmstad University, June 2012

Abstract

Modern vehicles are more electrical than mechanical. As the vehicle industry goes on, more mechanical parts are being replaced by electrical components, e.g. x-by-wire. Weight and production costs could be the major factors in this revolution. To ensure the safety and in time response (meeting request's deadline), the connection between these components is of great importance. Trying to find a protocol and standard that fits the vehicle industry to connect all ECUs, sensors and actuators together is not an easy task, since we are dealing with a system that has got a diverse type of traffic. Safety on the one hand and cost on the other hand are pushing the manufacturers to find the best networking protocol.

In this report I am going to investigate the possibility and the risk of implementing on-board diagnostics over Ethernet. UDS (unified diagnostic services) and DoIP (diagnostic communication over Internet protocol) are the protocols that are going to be studied.

To be more precise, I will investigate the possibility and the risk of UDS and DoIP implementation. The reason is that with the increasing number of ECUs and volume of data (parameter setting, downloading software, etc.,), the already implemented protocols and standards are not able to answer the required bandwidth. The wide acceptance and high bandwidth of Ethernet on one hand and its low cost infrastructure on the other hand has made Ethernet a candidate for this purpose.

The feasibility of Ethernet for in-vehicle network has already been investigated as I will show in the review of previous works. There are a couple of works that shows by using a suitable protocol in data link layer, we can meet the real-time requirement of a process. By reviewing these previous works we will find out that using Ethernet for vehicle on-board diagnostics is feasible.

After investigating risks and manufacturer's requirements along with worldwide legislations for the vehicle industry I will develop a DoIP gateway according to ISO 13400 which is connected to a CAN bus from one side and to a test tool via Ethernet from another side (Fig.1). UDS and DoIP protocols are implemented on both test tool and DoIP gateway. The practical part of the work is a complementary step in risk investigation that will be analysed.

Having Ethernet as a carrier allows us to make use of different protocols, based on different needs like bandwidth, latency and real-time characteristics. This fact allows the network to have multi-type data which is usually the case with in-vehicle network. That is, we can use Ethernet not only for on-board diagnostics but also for other in-vehicle domains, such as the chassis, infotainment and comfort.

At the end I will propose two models that will help the designers in this domain.

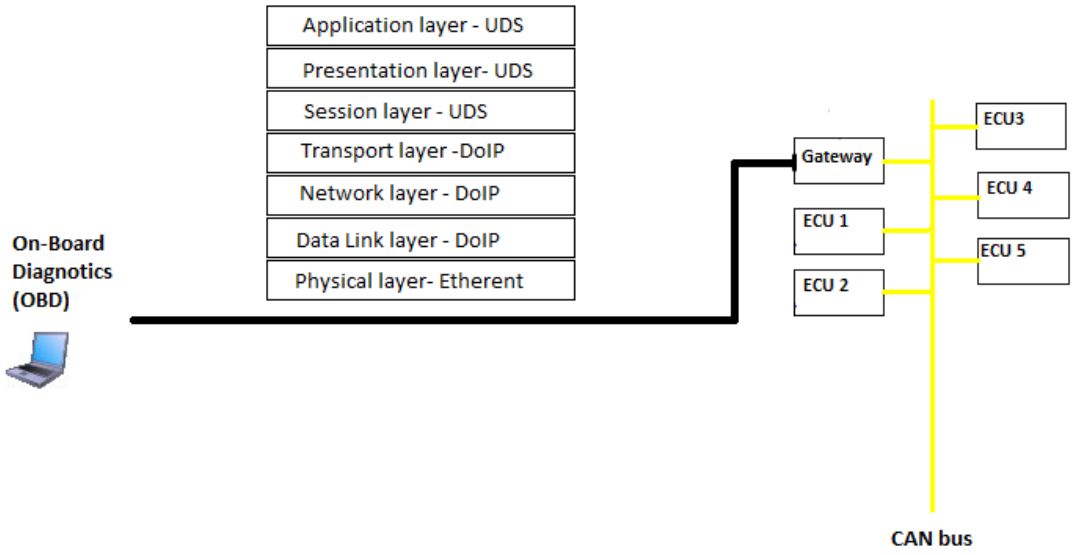

Figure 1. General View

Contents

1 Introduction

In this report I am going to investigate the possibility and the risk of implementing on-board diagnostics over Ethernet. UDS (Unified Diagnostic Services) and DoIP (Diagnostic Communication over the Internet Protocol) are the protocols that are going to be studied.

To be more precise, I will investigate the feasibility and the risk of UDS and DoIP implementation on Scania trucks. The reason is that with the increasing number of ECUs and volume of data (parameter setting, downloading software, etc.,), the already implemented protocols and standards are not able to answer the required bandwidth. The wide acceptance and high bandwidth of Ethernet on one hand and its low cost infrastructure on the other hand has made Ethernet as a candidate for this purpose.

Reviewing scientific articles and the practical world of industry I aim at revealing the important role of network communication in the vehicle industry while focusing on diagnostic domain and its need to have a more efficient network protocol. This can be studied mainly in chapter two. In chapter two I will also investigate communication protocols that are currently used in the vehicle industry.

According to the manufacturer specification and worldwide legislation that is reviewed in chapter three, I try to introduce my solution and briefly describe how it can solve the problem. Chapter three also covers DoIP as a candidate for our solution.

Chapter four makes the manufacturer aware of the obstacles that I faced in the implementation and also mentions the implementation details and the steps to a prototype by proposing two models. I close this chapter with a step by step description of the prototype that I achieved to implement and analyse its functionality.

Conclusion and future work in my last chapter (five) intends to help future designers.

1.1 Problem Statement (Motivation)

Currently on-board diagnostics of Scania's vehicle is done over CAN (Controller Area Network). Low bandwidth from one side and increasing number of ECUs and their data from other side are downgrading the system a little bit, that is, the current infrastructure cannot meet the new technology advancement, such as remote diagnostics, high bandwidth applications, applications in need of more shared resources, etc.. All these are happening at a time that a better QoS is a leading factor in the competitive market. The need for a higher download and upload speed at the time of repair in a repair shop and in the assembly line are among the possible use cases.

Due to the important role of on-board diagnostics in vehicle safety and comfort, new use cases and scenarios have been introduced that the old communication protocol is unable to answer.

We need to find a solution that can be implemented on the current products and infrastructure. DoIP is a candidate protocol but the question is how to implement this protocol in a way that is consistent with the manufacturer strategy and what type of risk we may face by implementing it. I will run both theoretical and practical investigations on how the system will improve upon DoIP implementation in relation to DoCAN (Diagnostics over Controller Area Network) and which parameter of the system will be affected.

In order to come up with the best implementation, we need to investigate the risk on how consistent the protocol is with the manufacturer requirements, which is going to be done by implementing some kind of prototype on the point to point scenario out of possible scenarios

mentioned in section 3.2. This prototype helps us to model the system in a better way by evaluating the related parameters.

1.2 Approach Chosen to Solve the Problem

I will propose different possible configurations for an in-vehicle network that can be used for implementation. Having chosen one of the configurations, I will propose two models as a result of my investigation and will implement a sample application using the models and come up with a comparison between the current diagnostics which is on CAN and the one on IP (Internet Protocol) . Due to some limitation which was imposed by current configuration of software architecture of the selected hardware, the sample application is close to my proposed models in a way that satisfy the general requirements of the specifications.

1.3 Thesis Goals and Expected Results

The following are the goals that I aim to achieve within the thesis scope:

1. Proposing a solution which is consistent with manufacturer strategy,
2. Implementation of a prototype according to the proposed solution.
3. A comparison on the efficiency of the system between diagnostics on CAN and diagnostics on IP.

2 Background

In this chapter, I provide the reader with some material on the domain of my research. A description of the related branch of science and listing protocols involved in this domain will give the reader a better understanding of the current work.

2.1 Networking and Communication

Networking is an important part in a distributed system, a system that is made of many components which are connected together to fulfil a task. Each component is responsible for one part of the system and in order to fulfil it, it needs interaction with other components. This requirement cannot be met without networking. In fact, by being able to connect components using suitable and reliable methods (protocols) we can make the system scalable and efficient.

While development of distributed systems; nodes with higher processing power, larger memory and more resource hungry applications, the support for distributed applications which need access to shared resources and memory allocation and in time responses, set a higher standard on performance to keep up with the requirement of the interactive applications on each node. So the new emerging standards and protocols are trying to have a better network performance, such as latency and data transfer rate and with a new approach to scalability, reliability and QoS (Quality of Service).

The basic need of a computer network is the packet-switching technique that enables packets targeted for different nodes to share a common communication link. Packets may undergo some delay that varies, depending on the time it takes to travel through the network to get to the target node. A packet may face several routers and switches on the path to the destination depending on the network type. Here communication is asynchronous, that is, packets are sent to a destination through different paths. If an outgoing link of a node is not accessible due to traffic, it will be queued in a buffer and will be sent as soon as the link is accessible again.

When we are talking about protocol, we mean a set of well-known rules and formats that are respected at the time of communicating between processes to fulfil a task. By respecting, it is meant that the sequence of messages that must be exchanged, the format of data in the messages and timing must be followed strictly. Usually a protocol is composed of some software modules that are located in the sending and receiving nodes.

In particular, a protocol suite is composed of three main sections: media, transport and applications. The interface between media and transport layer defines how the transport protocol makes use of a particular media (hardware) and the interface between application and transport defines how an application can make use of the transport layer. These are all implemented in the operating system.

Open System Interconnection (OSI) is a networking reference model that is composed of seven layers: application, presentation, session, transport, network, data link and physical layer. The OSI reference model illustrates a framework for the definition of protocols and not a specific suite of protocols. According to the OSI reference model, communication takes place between entities of the same layer in different nodes. Layers just communicate by the neighbouring layers by exchanging parameters and getting service at the Service Access Point (SAP) [1].

2.2 Vehicle Industry and Networking

Development in the electronic industry, replacement of more mechanical parts with electronic components and governmental legislation for more safety and less pollution are some of the driving factors of the vehicle industry towards vehicle network technology. Higher efficiency in performance, more comfort, lower costs and shorter time in manufacturing and as a result getting more cost effective, are some of the advantages of this approach. Looking at early stages of the vehicle industry, one can say that a radio was the only electronic part in a vehicle, but now we are dealing with a system that is more or less electrical. The braking system, navigation and so many other control units for doors, windows, heating, safety, security, etc. are all electrical. When we are talking about electrical control units, we mean a unit that needs inputs from sensors and output to the actuators. The process of getting input and giving output are done by exchanging data between units that is needed for vehicle operation. Suppose the engine is going to inform the transmission part of the engine speed, and the transmission part is going to inform other parts about the instance of the gear shift. All this process needs to be done in real-time, that is, a quick and reliable response is needed. This example makes the role of the network as a medium for exchanging data more clear.

With the growth of electronic control units in vehicle and the requirement to be connected together, more cable was being used which would make the system more complex. The complexity was not just due to number of wires, removal or adding a new component required a change in design which would make the process difficult. To deal with this complexity, the manufacturer decided to use a central bus and the main electronic units were connected to the main bus. This allowed different units to be plugged and unplugged from the network easily, which leads to a more flexible and faster manufacturing process. The main bus is a special communication network that connects the main units. Each unit is a node that is controlling a set of components for a certain task. The communication and exchange of data between all these units and components are based on a certain protocol and standard.

The reason for introducing new standards and protocols is a characteristic of the vehicle environment: external noise immunity, operation in a harsh environment and being robust and reliable as well as cost effective, when we are dealing with a large production scale. In the following section, we will consider different protocols and standards that have emerged in this industry.

It should be mentioned that many communication protocols have been defined in the automotive industry covering OSI layers; here I mostly go through those involving physical and data link layers.

2.2.1 Controller Area Network (CAN)

Controller Area Network (CAN) is field bus technology that was mainly developed for the automotive industry to interconnect embedded applications for data exchange. It is capable of providing a data rate of up to 1 Mbps, theoretically. It was a solution for the complexity of wiring in the automotive industry which has high electrical interference immunity and is fault-tolerant. These features made CAN popular in the automotive, marine, medical, manufacturing and aerospace. CAN protocol is a message based protocol and nodes compete to get the bus based on arbitration. In fact the priority is assigned based on arbitration and a node with higher priority is allowed to use the bus and no other node is allowed to preempt at the time of transmission [2]. This type of priority assignment lets a node win for a long time and causes the other node to lose their deadline. CAN is a multi-master serial bus that transmits data by broadcasting on a shared bus, according to the OSI architecture model. It is

a protocol for physical and data link layer that is defined in ISO 11898. Table 1 lists all tasks carried by a data link layer and a physical layer.

Table 1. Layered Architecture of CAN - ISO 11898

Data link layer
LLC (Logic Link Control) Acceptance filtering Overload notification Recovery management
MAC (Medium Access Control) Data encapsulation/ decapsulation Frame coding Error detection Error signaling Acknowledgement Serialization/ deserialization Medium access management
Physical layer
PLS (Physical Signaling) Bit encoding/decoding Bit timing synchronization
PMA (Physical Medium Attachment) Driver/receiver characteristic
MDI (Medium dependent Interface) Connector

Table 2. CAN Versions

NOMENCLATURE	**STANDARD**	**MAX.SIGNALING RATE**	**IDENTIFIER**
Low-Speed CAN	ISO 11519	125 kbps	11-bit
CAN 2.0A	ISO 11898:1993	1Mbps	11-bit
CAN 2.0B	ISO 11898:1995	1Mbps	29-bit

SOF	11-Bit Identifier	RTR	IDE	r0	DLC	0…8 Bytes of Data	CRC	ACK	EOF	IFS

Figure 2. Standard CAN Frame

S O F	11-Bit Identifier	S R R	IDE	18-Bit Identifier	RTR	r 1	r 0	DLC	0…8 Bytes of Data	CRC	ACK	EO F	IFS

Figure 3. Extended CAN Frame

On a CAN bus, a logic high is zero and known as dominant and a logic low is one and known as recessive. That is why the transmitter's input and receiver's output are internally pulled up. In CAN we deal with message frame that their payload are maximum eight bytes, these short messages are broadcast in the entire network which causes data consistency among nodes in the system.

The access to the bus is event-driven and random, which is managed by bit-wise arbitration. The MAC protocol is CSMA/CD but it takes a more systematic approach which is inherited from the CAN signaling characteristic (CSMA/CR). The frame has got an identifier which is used not only for priority purposes but also helps the receiver to filter the messages. In fact each node has got a unique bit pattern that by the help of mask registers, acts as a filter. This characteristic of CAN makes it attractive for real-time application. Each node will check if the bus is idle before starting to transmit. Nodes which are going to access the bus will contend for the access simultaneously, the higher the priority the lower the binary message identifier. Nodes that are sending their identifier listen to the bus at the same time. If one node sends a recessive bit and another one sends a dominant bit, the one with dominant will win and the other one will stop sending the identifier. It is a kind of AND operation. Since the dominant bit overwrites the recessive bit, and the node monitors its own transmission by listening to the bus, it can see if the bus has got the same value as it has sent or not, if so it will continue with the rest of message frame. Table 2 shows different versions of the CAN protocol.

Figure 2 and figure 3 illustrate the message format of the CAN frame both for the standard and extended version.

There are four types of messages that can be sent over a CAN bus:

- Data Frame: this is the most common message type which is made of an arbitration field, data field, CRC field and ACK field. For this message type the RTR bit is dominant and the data field can vary from 0 to 8 bytes. The length of the data part is defined in DLC parameter, which is 4 bits long and any value larger than 7 indicates 8 bytes. The CRC filed is 16 bits long. On a reception of a correct message, the receiver overwrites the recessive ACK bit by a dominant bit so that the transmitter will find out if the data needs to be retransmitted.
- Remote frame: this message type is similar to data frame with two differences: there is no data and the RTR value is recessive. It is used to request transmission from another node.
- Error Frame: if a node detects an error in a message, it will start sending this frame consisting of six recessive or dominant bits, followed by an error delimiter, which makes other nodes to detect the error. There is a counter mechanism which does not let a faulty node make the bus busy by switching the node from active to passive and bus-off mode [3]. There are five methods for error checking that are at two levels: message level and bit level. On the occurrence of any of these errors the message needs to be retransmitted, figure 4. TEC stands for transmit error counter and REC stands for receive error counter.

- Overload frame: this frame is similar to error frame and is sent by the overloaded node to request some delay between messages [4].

The detail of the protocol shows that CAN has not been designed for large blocks of data to be sent in a point to point mode.

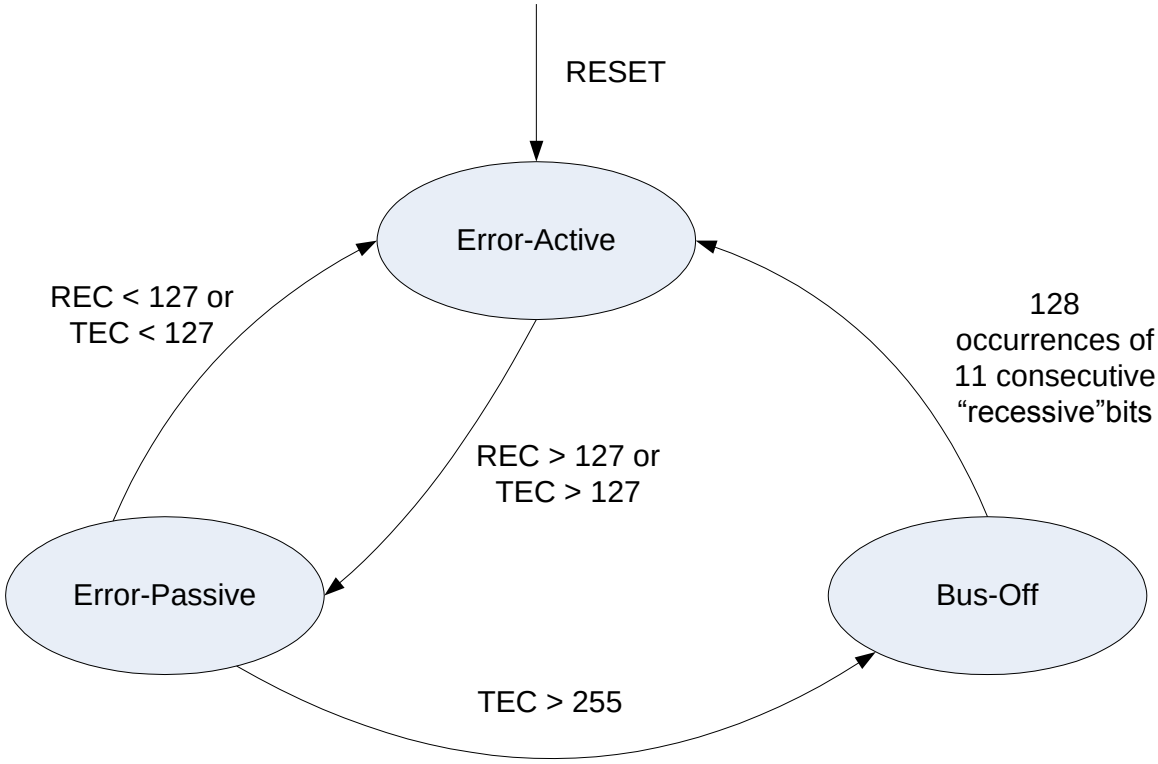

Figure 4. State transition Diagram of a Node

2.2.2 FlexRay

Being both time-triggered and even-triggered has made it a good solution for a safety-critical application. It is mainly intended for high speed control application in the automotive industry. It also supports single and dual channel communication. In order to be fault tolerant it is supposed to be in dual channel mode that makes it in redundant configuration. It provides a bus guardian at the physical interface, which leads to error containment and error detection. FlexRay is really flexible regarding network topology and support for redundancy. A network architecture can be configured in a bus or a star or a multi-star and it is not mandatory for all nodes to be in replicated channel mode or having a bus guardian. The communication cycle is made of a static segment concatenated by a dynamic segment, symbol window, network idle time and the total communication cycle is set statically at design time. The static segment is based on the time division multiple access (TDMA) scheme. Each node can access the bus for a number of equal size static slots and these time slots can be used just by the defined node and will be unused if the node has got nothing to send on the bus. After the static segment, the dynamic segment starts. The dynamic segment is event-triggered and is based on the flexible TDMA (FTDMA) scheme. The communication in this segment is based on mini slots. Each node can access the bus for a dynamic number of mini slots.

In FlexRay each frame consists of three parts: header, payload and trailer segment. The header is made of frame ID, payload length and a CRC. The CRC is over frame indicators, frame ID and payload length. The trailer segment includes a CRC over header segment and payload segment which is 24 bits. The data is carried in the payload segment and it can be 0- 254 bytes. Frame ID has got two definitions, based on being used either in a static or dynamic segment. In the static segment it defines the slot position and in the dynamic segment it defines the priority of the frame. Based on this priority, in dynamic segment a node can access the bus sooner [5].

Since in static segment it is possible to assign as many slots as needed for each node, it is more flexible than the TTP/C MAC protocol. In the automotive industry, where both critical and non-critical applications exist, the flexibility of FlexRay both in topology and replicated channels is of great use regarding cost and complexity [6].

2.2.3 Media Oriented System Transport (MOST)

The physical layer in this protocol is different from others and it is a plastic optical fibre (POF) which has got a better resilience to electromagnetic interference (EMI). Media Oriented System Transport (MOST) was mainly developed for in-vehicle multimedia and infotainment. The current MOST generation is the third one and it has a data rate of 22.5-150 Mbps with a QoS specified by the standard.

There are three different types of message in MOST: control, stream and packet message channels, which are provided by the protocol stack (Network Service) that the standard has specified [7]. The stream channel type is used for multimedia data and the packet type is used for TCP/IP-like data. These two channel types use more bandwidth compared to the control channel which is a function-oriented one. All the control operations on each device in MOST networks are defined as Function Block (FBlocks) which lets a specific operation be triggered on the device. FBlock ID, instance ID, function ID, operation type, user-defined parameters and telegram information are the elements of a control message. The functions can be either attribute or method, like an object oriented programming [8].

There is a base signal that is provided by a master to make all the nodes on the bus synchronous in time.

MOST bus includes all 7 layers of an OSI model; physical layer, data-link layer, network layer, network service layer 1, network service layer 2 and application layer [9].

2.2.4 Local Interconnect Network (LIN)

Cost and speed have been always the dominating factors for choice in networking technology. At field level we are more concerned about controlling actuators and reading data from sensors. In this level, we are dealing with mechanical component and low speed requirement.

Low computational power, small amount of memory and low cost oscillators along with the simplicity of communication controllers, make the low cost objective possible for body domain in the vehicle industry. The communication is based on a master-slave mechanism. Each cluster in LIN is consisting of one master and some slaves connected to a common single serial bus. Due to EMI limitation, it is not possible to achieve a data rate higher than 20kbps.

LIN is a time-triggered network and the master node manages the message transmission by a schedule table. The table contains a list of frames and their frame slots which ensure the determinism in the sequence of transmission. The frame of data that can contain up to eight bytes of data is broadcast by the master and all the nodes can listen and the one which possesses the identifier of one byte length can fill the response field. The identifier is made of a six-bit address field and a two-bit check field [10]. If there is no update for the requested data, there will be no response, which leads to bandwidth saving. By providing sleep mode in nodes, energy saving is also provided in this standard.

There are five frame types in LIN: sporadic, unconditional, event-triggered, diagnostic and user defined [6].

2.2.5 Byteflight

Byteflight is designed for safety critical automotive application that can provide a speed of 10Mbps. Having optical fiber as its physical layer provides resistance to EMI effects. Each node in the network has got a unique identifier. Bandwidth is reserved for each node by a kind of time-triggered mechanism. Each node has got a slot counter that is restarted upon reception of a synchronization pulse from a SYNC master node. Nodes start counting upon reception of each mini slot and if a node's slot counter is the same as its identifier then it is the time for that node to access the bus. During the message transmission the counter will pause, so there will be no mini slot missed. There is a possibility that low priority messages miss their deadlines. The duration of a mini slot is 250 us and the mechanism can be considered as flexible time division multiple access (FTDMA) [11].

In fact, low jitter for high priority messages is guaranteed, while bandwidth allocation for low priority messages is flexible.

As industry goes on, the time for production is shortened and it is not possible to consider everything at the early stage of design, the possibility to add additional functionalities which are not going to affect the primary functionality is of great importance which can be done by introducing low priority messages in the system. This issue was one the requirements that was considered at the stage of designing the protocol. The data frame in Byteflight consists of a start sequence, an identifier field, a field indicating the length of data segment, data segment and CRC field in the mentioned order. The data field is 12 bytes and start sequences are six dominant bits. The synchronization of receiver and transmitter at the time of data transfer is done at byte level, i.e. each byte from identifier to CRC starts with a recessive bit and stops

with a dominant bit, which provides a better mechanism for reducing jitter by a fixed length frame, compared to CAN [12].

2.2.6 TTP/C

TTP/C is a time-triggered protocol designed for SAE class C applications. SAE class C covers real-time control like a power train control, vehicle dynamics, brake by wire, etc. TTP/C is used both in the automotive industry and avionics. In TTP/C network, the nodes are connected by two replicated channels. Network topology can be either star or bus.

Being based on TDMA protocol and using bus guardian, no node can access the bus out of its predefined time slot in each round. That is why TTP/C is a deterministic protocol [6]. The communication schedule is stored as a message descriptor list in the communication controller of each node. A round in TTP/C is formed by a sequence of slots where each node sends at most two frames, one on each channel. A number of TDMA rounds form a cluster which repeats itself in a cycle.

Cold start frame, data frame with explicit C-state and data frame with implicit C-state are three types of the frames that TTP/C deals with. Each frame is capable of carrying 240 bytes of data at a speed up to 25Mbps. TTP/C resolves the hypothetic faults in messages by some defence mechanisms: membership services, redundancy management, rapid mode changes, fail silence to avoid error spread and bus guardian for faulty transmitters. In fail silence mode, the network ensures either a correct message in value is received, or no message at all is received by a receiver. The membership service helps all the nodes to have some knowledge of other nodes in the network and the state of the network [13].

2.2.7 TTCAN

Time-triggered CAN is a communication protocol placed at the top of data link layer and physical layer of the CAN protocol. The bus topology, frame format, transmission support and data rate are as CAN protocol states, but the controllers are not supposed to retransmit upon transmission error, also nodes are not allowed to transmit out of their system matrix (SM) which helps to a better real-time performance network. SM is a predefined and fixed time schedule that is used during the time-triggered window and is repeated.

Like FlexRay, each basic cycle is formed by some time-triggered windows followed by an event-triggered window. During an event-triggered window, access to the bus is allowed based on priority as in CAN. A sequence of basic cycles makes a system matrix which has a fundamental role in real-time performance of the network. A reference message broadcast by the time master indicates the start of a cycle which synchronises all the nodes [6] [14].

2.2.8 SAE- J1850

It is an SAE class B protocol that its MAC layer is based on priority, designed for networks with non-stringent real-time requirements like diagnostic domain or body domain. It has two variants: 10.4 Kbps single wire with variable pulse width modulation and 41.6 Kbps two wires with pulse width modulation. Both variants are CSMA/CR protocols and collisions are handled like CAN, by arbitration [6]. This contention bus is inherited from an open architecture that lets node be added or removed from a network without much impact on the rest of the system which is called non-destructive contention resolution. Short message length, short latency, payload of 12 bytes and master-less network are some of the characteristics of J1850. In this protocol, each frame is composed of a start frame, header, data field, CRC, end

of data (EOD), in-frame response (optional) and end of frame (EOF). This SAE standard specifies a set of communication requirements that makes a network cost effective, flexible and simple for complex applications targeted by the automotive industry [15].

2.2.9 Ethernet

One of the most common local area networks in industry is Ethernet (IEEE802.3) with a bit rate of up to 1 Gbps. It was designed for an office environment, but able to be deployed at field level by some EMC protection methods like USDT (unshielded twisted single pair) [16]. Making use of advanced techniques in traffic shaping, flow control and congestion control that usually happens in an office environment due to volume of data and number of users makes it a good candidate for the current vehicle industry, which is facing an increase in the number of network nodes and volume of data [17].

Ethernet was originally designed for computer networks where the bandwidth and cost of hardware are more important than jitter and guaranteed deadline. Several works have been done on its deployment for real-time application in the automotive industry [18]. Due to its large scale use, the production cost of the Ethernet is noticeably low, which is of great importance for vehicle manufacturers [19].

Today's vehicle is a system carrying different features driven by several dozens of electronic control Units (ECU). They are often interconnected by communication protocols like CAN, MOST, FlexRay and so on. There are recently some models of vehicles which have made use of Ethernet in providing high bandwidth and low cost in domain like diagnostics, infotainment, ECU firmware update and etc. It seems among all manufacturers, BMW has a leading role in Ethernet deployment in its products as can be seen in recent series.

There have been several works to deploy Ethernet for different vehicular domains, both real-time and non-real-time applications. Being able to provide high bandwidth with low cost equipment is a great advantage that Ethernet brings to the users [20]. All these works have been done due to the existence of several data types in vehicle, which requires different QoS regarding bandwidth, latency, cost of equipment and guaranteed jitter. It seems Ethernet could be a good candidate for carrying several data types.

Each data type needs a certain QoS. Currently, in-vehicle communication is composed of several data types which makes the networking more complex and costly for the designer and the manufacturer. Transmitting each data type on the proper network protocol or all on one common protocol is a question which has instigated a lot of research work. The deployment of different network technologies leads to an inflexible architecture and makes the cable harness complex and costly. [20] Introduces IP as a networking protocol that is perfectly suitable for having applications that are independent of different network technologies. Different transport protocols can be implemented on higher layer that can use IP characteristics. IP characteristics allow it to be implemented independent of a physical and MAC layer. It also reviews Ethernet as a dominant network technology in computer networks.

Being able to communicate in full duplex mode has improved Ethernet capability which makes it collision free [21]. There has been a lot of work done to make Ethernet suitable to provide appropriate QoS and real-time guarantees for different applications [50] [51]. Some of these approaches are using deterministic mechanisms like time-division multiple access (TDMA) over Ethernet, that provides guaranteed access to the network for each application [22]. Some are based on a time-triggered mechanism, but they need some changes in the hardware side to support a better network access control [23]. Some of the works have focused on assigning priority to different data types, by tagging them as defined in IEEE

802.1p. Some of the approaches have their points in cost and some in complexity [20]. There are also several works which have focused on the topologies like tree, daisy chain and ring and how different types of traffic can be managed better, by utilizing a different topology on a network that is using Ethernet as its physical layer and MAC layer, compliant with IEEE802.5 [24] [25] [27] [28].

[29] Has investigated the advance driver assistance systems (ADAS) traffic on different topologies under realistic in-vehicle traffic and by prioritizing packets in application level and not in ECU level. The first topology has been designed to minimize the cost by reducing the number of switches and using existing components. The second topology aims at improving service performance by distributing network load with the help of an extra switch and in the last topology the goal is to decouple the traffic of a transmission domain and assistance domain by introducing a daisy chain topology using extra switches.

Considering the seven layers of the OSI model, Ethernet is concerned about the layer 1 and 2 (physical layer and data link layer). In the OSI model, the lower layers deal with how bits are moved around whereas higher layers deal with security, reliability and data transmission.

In fact, Ethernet is considered as a truck that can carry stuff. Different protocols like UDP, TCP, SNMP and many more are implemented on top of Ethernet that are on higher layers in the OSI model. These higher layer protocols are trying to implement security and reliability. Ethernet protocol states the frame length and its segments and deals with signaling protocol and medium access control (MAC). Protocols implemented on Ethernet try to make it more reliable and secure and improve its efficiency by different traffic shaping, task scheduling and encryption algorithms.

Frame, media access control protocol, signaling components and physical medium are the four building blocks of Ethernet protocol which are described as follows:

- ✓ Frame: It is the packet consisting of bits that includes header, addresses, data type and CRC segment. The data segment can be from 46 to 1500 bytes.
- ✓ Medium access control protocol: States the rules that each node should follow in order to access the shared media in a fair manner, so each node is going to have equal right to access the media. Ethernet nodes use CSMA/CD to access the media.
- ✓ Signaling components: Are the electrical components that are involved in receiving and transmission of signals over channels.
- ✓ Physical media: Includes the cables and hardware that carry digital signals between nodes [30].

2.2.10 A Comparison among protocols

Each described network protocol and technology has got its own characteristic; some are preferred over the other because of the cost of implementation and equipments, some more safety and error protection and some are providing higher speed. You can see that some manufacturers prefer to utilise not only one but a combination of some of the protocols for different domains concerning in-vehicle communication, based on the need of that domain for high bandwidth, low cost and real-time requirements. On the other hand some manufacturers rather stick to one network technology for all the in-vehicle domains with just some

modifications on some parts in order to escape the complexity that multi-protocol communication network may bring.

Most of the protocols use message broadcasting which introduces more traffic to the network but it lets all the nodes know the state of the whole system at each instant.

A comparison among CAN, Byteflight and TTP/C shows that all three use a CRC with a hamming distance of 6 [14] which means the probability of not detecting an error is more or less the same. As in CAN, we are going to have a retransmission of messages with error, which results in delay and if the node is going to contend for network access this delay is longer.

It is possible to categorise all networking technology based on medium access strategy. CSMA and TDMA are the most common. One provides high bandwidth utilisation but not deterministic behaviour, whereas the other one has got guaranteed jitter and deterministic behaviour for the system [14].

The author in [10] compares LIN and TTP/A and introduces them as a field protocol and believes that LIN focuses on the basic functions needed for real-time communication, whereas in TTP/A it will go one step further and design an architectural framework for on-line diagnostics. In TTP/A plug-and-play capability is known as its main objective.

The main problem with Ethernet is its incapability in providing real-time behaviour, i.e., guaranteed response time and determinism. The CSMA/CD mechanism that is used in Ethernet is not a deterministic method and under collision a node starts to wait for a random amount of time to get access for transmission. Some researchers propose the full-duplex Ethernet as a solution, but the problem that usually arises is the unrestrained buffer contention in the switches and the guaranteed upper bound latency on packet transmission. There are several works done on Ethernet deployment in a real-time environment, by proposing protocols like AFDX, TTEthernet and Ethernet AVB [38].

There are several higher layer protocols that focus on data segmentation, improving message content that is transferred between ECU, start-up time like J1939 and OSEK.

Table 3 provides a brief comparison among some of the described standards:

Table 3. Protocol Comparison

	General	**Network bandwidth**	**Network topology**	**Scheduling**	**Fault/tolerance and additional services**	**Synchronization**	**Functional domains**
LIN	- low-speed - low-cost - time-triggered	20 kb/s	- bus	- master/slave - polling list based on schedule table	- collision resolution by master node - bandwidth and energy saving services	Synchronization of nodes by the 'Sync' field in a LIN message frame sent by master node	- Body/comfort
CAN	- low-cost, simple - twisted pair - event-	Up to 1Mb/s	- bus - star	- CSMA/CR - Bitwise arbitration based on message	- CRC - automatic retransmission - error counter and	Bit synchronization	- Body/comfort - Powertrai

	triggered - de-facto standard - most widely used			identifiers	bus-off state schemes - Additional fault tolerance services are necessary on upper layers		n - Chassis
ByteFlight	- hybrid paradigm - POF	10 Mb/s	- star	- FTDMA based on message identifiers - master/slave (for synchronization)	- star coupler (to avoid 'babbling idiot') - CRC	Synchronization of nodes by the 'synch pulse' sent by master node	- Passive safety - Safety-critical applications
TTP/C	- twisted pair or POF - time-triggered	Up to 25 Mb/s (depends on network topology)	- bus - star	- TDMA - predefined and fixed communication schedule (MEDL)	- replicated channels/nodes - star coupler (star topology) - CRC - bus guardian - membership function - clique avoidance algorithm - error containment mechanisms - never-give-up (NGU) strategy - mode change management (different schedules)	Distributed clock synchronization	- x-by-wire - Chassis (active safety)
TTCAN	- low-cost, simple - twisted pair - hybrid paradigm - time-triggered layer on CAN	Up to 1Mb/s	- bus - star	- TDMA in exclusive windows, CSMA/CR in arbitration windows - predefined and fixed communication schedule (system matrix) - master/slave (for synchronization)	- CRC - mode change (CAN to TTCAN and vise versa) by master node - Additional fault tolerance services are necessary on upper layers	*Level* 1 and *Level* 2 time synchronization by *reference message* sent by master node	- powertrain - chassis - x-by-wire - safety-critical applications

FlexRay	- hybrid paradigm - twisted pair (bus) or POF (star) - future de-facto standard - can be used in two modes (time or event-triggered)	10 Mb/s	- bus - star - multi-star	- TDMA in the static segment, FTDMA in the dynamic segment - predefined and fixed communication schedule (elementary cycle) - master/slave (for synchronization)	- scalable dependability [1] - dual channel redundancy (optional) - CRC - never-give-up (NGU) strategy - bus guardians for only time-triggered traffic (optional) - trigger monitoring	Distributed clock synchronization	- powertrain - chassis (active safety) - x-by-wire
MOST	- cost-effective - data-efficient - hybrid paradigm - de-facto standard for multimedia/infotainment - POF	25 Mb/s	- ring - star	- master/slave - support for (a)synchronous - point-to-point video and audio data transfer	- support for “plug and play” - support for multiple master nodes	master node based synchronization by sending the preamble	- multimedia - infotainment

2.3 On-Board Diagnostics (OBD)

On-board diagnostics or OBD provides diagnostic functionalities in both emission and non-emission related functions of vehicles, which assists repair technicians to identify the malfunctioning in both emission systems and non-emission systems. In fact, OBD standards provide an interface to vehicle internal states for the maintenance technician. It has evolved over years and now its objectives in non-emission related functions includes software updates of ECUs and collecting diagnostic trouble codes stored on these ECUs. With the increase in number and complexity of electronic control units, the time and effort to find a defect in vehicles is going to be prolonged which makes the role of new technologies and protocols in vehicle diagnostics more clear. In other words the difficulty in solving the problems increase exponentially with complexity [31]. There are several protocols in this field that specify how the regulations should be followed. Protocols and standards [32] related to on-board diagnostics helps the designer to provide a better interface so that a technician can find out the internal state of the vehicle more easily. Protocols define the procedure and requirements that the designer of the test tool is supposed to follow for its proper functionality.

On the proper connection of a diagnostic tool to a vehicle, it is possible to read trouble codes, updating firmware of an ECU and setting parameters. Worth mentioning, the main objective of OBD has been controlling the vehicle air-pollution level as can be seen in OBD-I and OBD-II but in the current study I am not dealing with the main objective of OBD and I focus only on the general concept of OBD or better said the diagnostics.

As the number of ECUs and the volume of data are increasing over time due to governmental regulation for DTCs and manufacturer requirement, the old networking protocols are no longer suitable. The required time for downloading DTCs or firmware is more than acceptable. Just imagine how long it takes to download 80MB of data with a 250kbps CAN connection. It seems like the introduction of Ethernet into the vehicle industry is going to be a breakthrough to solve this problem. There are also many works that are considering on-line on-board diagnostics due to its importance to OBD [33].

Reports from vehicle technical support shows that more than two thirds of the time is spent on finding the fault and just a short time is spent for the repair. Most of the current diagnostic tools that are available on the market are from the handheld class which suffer from: relatively unchangeable hardware, low transmission speed, not being able to upgrade the tool easily and low processing power and because of limited memory it is not possible to implement intelligent fault diagnosis and troubleshooting [35].

There are several international standards like ISO and SAE that define the common requirements of diagnostic systems based on the OSI model. Some protocols are dedicated to just one layer and some are a protocol suite that covers all layers. Those for a specific layer are regardless of higher and lower layers, i.e. they are independent of neighboring layers. The services described in these protocols are implemented in various applications like road vehicles, tachograph systems, data communication between towing and towed vehicles and mainly diagnostic applications.

There are mainly two series of protocols KWP2000 [32] and UDS [36] apart from emission related legislated protocols [37]. These two protocol series define the format of services, command codes and required steps for a diagnostic session

Tables 4 and 5 show some of the protocols that are used for OBD services [35] [36]. Table 4 lists all OBD protocols in different layers when KWP2000 is in application layer whereas

table 5 lists all OBD protocols when application layer is based on UDS. Table 5 also includes OBD emission related standards.

Table 4. Applicable Protocols to OBD and KWP2000

OSI 7 layers				
Physical (layer 1)	ISO 9141-2	SAE J1850	ISO 14230-1	ISO 11898 ISO15765-4
Data link (layer 2)	ISO 9141-2	SAE J1850	ISO 14230-2	ISO 11898 ISO 15765-4
Network (layer 3)				ISO 15765-2 ISO 15765-4
Transport (layer 4)				
Session (layer 5)				ISO 15765-4
Presentation (layer 6)				
Application (layer 7)	SAE J1979/ISO 15031-5	SAE J1979/ISO 15031-5	SAE J1979/ISO 15031-5 ISO 14230-2	SAE J1979/ISO 15031-5

Table 5. OBD Protocols and UDS

<table>
<tr><td>Applicability</td><td>OSI 7 layers</td><td colspan="4">Enhanced diagnostics services</td><td>WWH-OBD</td></tr>
<tr><td rowspan="8">Seven layer according to ISO/IEC 7498-1 and ISO/IEC 10731</td><td>Application (layer 7)</td><td colspan="4">ISO 14229-1, SIO 14229-3(UDSonCAN), ISO 14229-4(UDSonFR), ISO 14229-5(UDSonIP), ISO 14229-6(UDSonK-Line), further standards</td><td>ISO 27145-3</td></tr>
<tr><td>Presentation (layer 6)</td><td colspan="4">Vehicle manufacturer specific</td><td>ISO 27145-2</td></tr>
<tr><td>Session (layer 5)</td><td colspan="5">14229-2</td></tr>
<tr><td>Transport (layer 4)</td><td rowspan="2">ISO 15765-2</td><td rowspan="2">ISO 10681-2</td><td rowspan="2">Not applicable</td><td rowspan="2">ISO 13400</td><td rowspan="4">ISO 27145-4</td></tr>
<tr><td>Network (layer 3)</td></tr>
<tr><td>Data link (layer 2)</td><td rowspan="2">ISO 11898-1, ISO 11898-2
(CAN)</td><td rowspan="2">ISO 17485
(FlexRay)</td><td>ISO 14230-2</td><td rowspan="2">ISO 13400
(IP)</td></tr>
<tr><td>Physical (layer 1)</td><td>ISO 14230-1
K-Line</td></tr>
</table>

3 Requirement and Solution

The current chapter will investigate the manufacturer's requirement and then will propose a solution in the second section, inline with the studied requirements in the OBD domain.

3.1 Scania network and DoCAN Requirements

Scania's vehicle network is based on CAN. Scania's ECU CAN requirements are divided into a functional and a physical specification. As can be seen in figure 5, the network is divided into three buses (sub-networks): red, yellow and green. The red bus is the most critical bus and the green is the least. The baud-rate on the red bus is 500 kbps and delivers the transmission domain's data that are related to engine, gearbox and brakes. The three buses are connected together by a coordinator which isolates the buses from each other's traffic and causes a better level of safety [45].

As mentioned earlier, the CAN protocol takes care of the first two layers of the OSI protocol suite. The implementation of other layers to fulfil communication between ECUs is done based on J1939 which the Scania network is based on [48], figure 6. J1939 focuses on message format, length of payload which is always 8 bytes, start-up procedure and a few more cases. It was basically designed to handle the communication between engine, transmission and brake but, later was extended to other application domains. This protocol was proposed by the Society of Automotive Engineer (SAE).

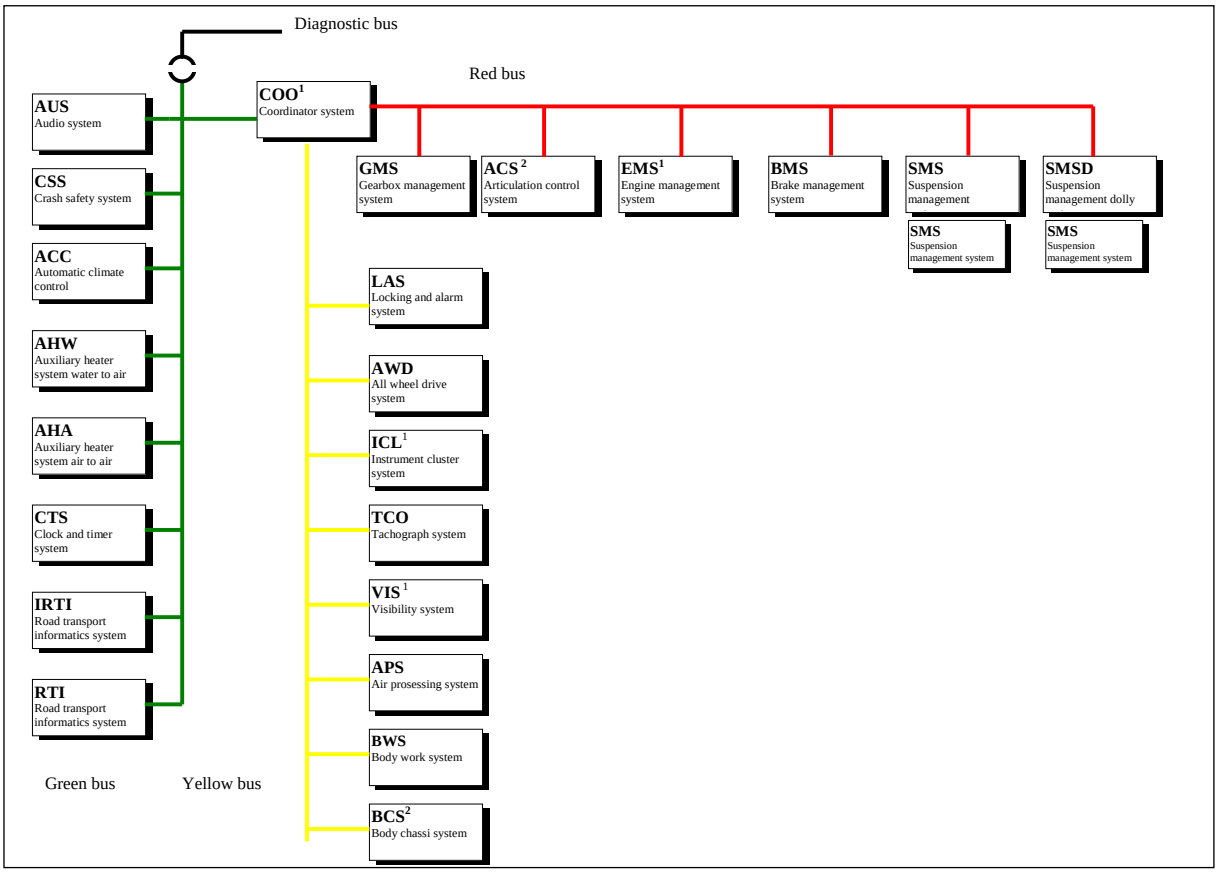

Figure 5. Scania's CAN Network

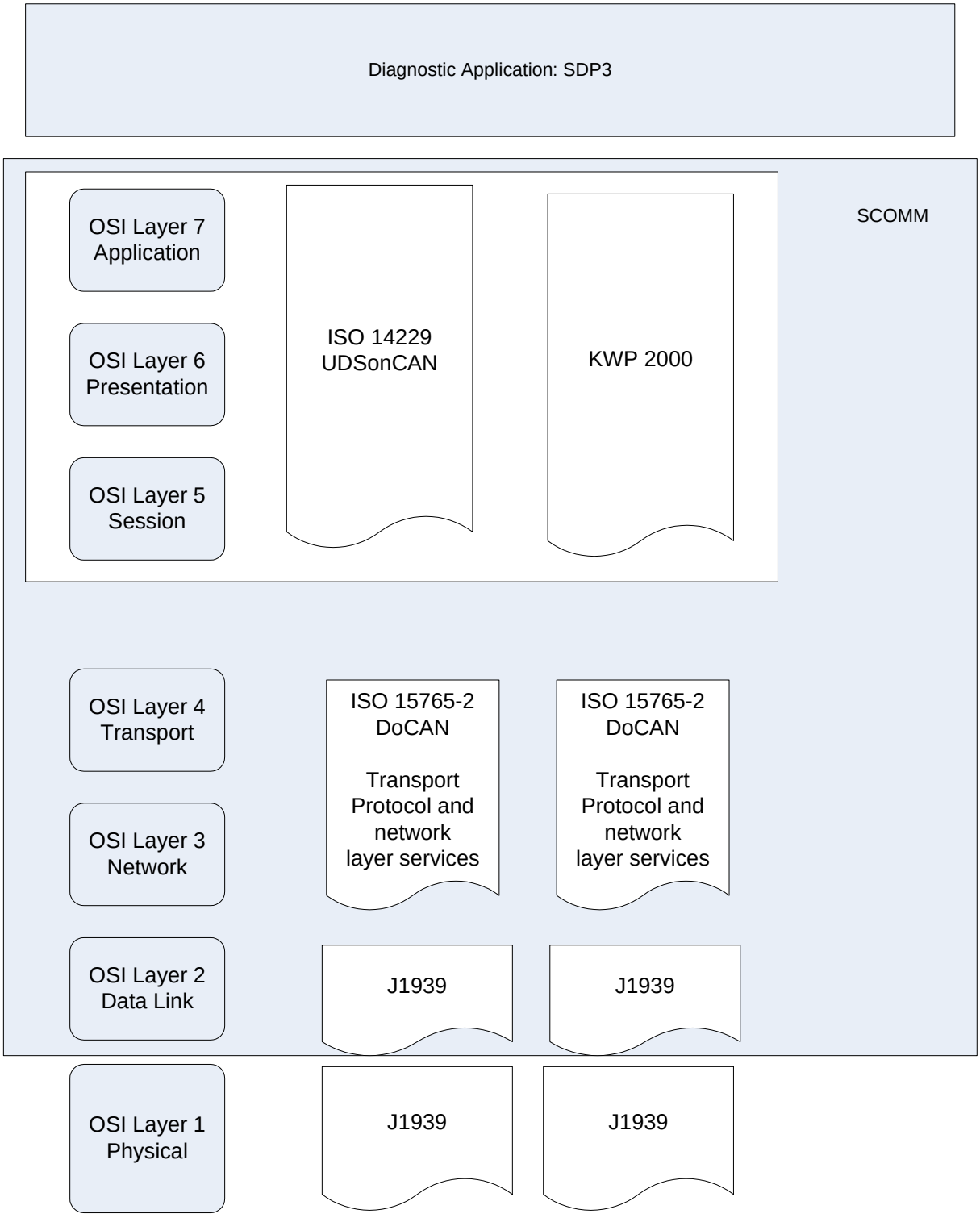

Figure 6. A View of Network Layers of Scania's Diagnostic Application

In the OSI layer, a data link provides a mechanism so that nodes can interact in an efficient way. It provides a mechanism for sharing data and process output and allows systems and subsystems to interact with each other, which are the main objectives of J1939 and is used in the vehicle industry.

SAE J1939 is a high speed communication network aiming at real-time closed loop control functions between ECUs distributed all over the vehicle. It provides all the functions of J1708/J1587 that is an old widely used network with lower speed. J1939 is based on CAN and uses 29 bit identifier (CAN extended format). SAE J1939-21 provides the possibility that both 11 bits identifier and 29 bits identifier coexist on the same network without any interference [39].

Table 6 shows how address information parameters are mapped into the CAN frame when the data link layer is according to SAE J1939. Note that the table only is related to normal and physical addressed messages [40].

Table 6. ISO 11898 and SAE J1939

SAE J1939 name	P	R	DP	PF	PS	SA	Data field
Bits	3	1	1	8	8	8	64
Content	Default 110(bin.)	0	0	218(dec.)	N_TA	N_SA	N_PCI, N_Data
CAN Id bits	28-26	25	24	23-16	15-8	7-0	-
CAN data byte	-	-	-	-	-	-	1-8
CAN field	Identifier						Data

P: Three bits of the priority field optimise the message latency in the CAN bus

R: A reserved bit.

DP: Data page.

PF: Protocol data unit format, it is used in connection with data part. The value is related to the type of addressing, functional, physical, mixed or normal fixed addressing.

PS: This field contains the target address.

SA: This field contains the source address.

N_Data: It refers to the data part.

N_PCI: It represents the type of the protocol data unit; single frame, first frame, consecutive frame or flow control.

ISO 15765 has been designed to define the common requirements of vehicle diagnostic systems implemented over CAN as described in ISO 11898. According to the OSI model, the application layer is responsible to provide services for diagnostic application.

The diagnostic application in Scania's products is based on ISO 15765, DoCAN, diagnostics over CAN which defines the requirement for physical layer, data link layer, network layer and transport layer of the OSI model. The Application layer is defined by either KWP2000 or ISO 14229 (UDS) [36] [46] [47]. Since UDS is known to be better than KWP2000 regarding clear specification and some technical issues, it is intended to have all ECUs on UDS rather than KWP2000. It should be mentioned that an ECU is either KWP2000 compatible or UDS compatible. The diagnostic application that is used as user interface is SDP3 and the whole protocol suite is provided by a package called SCOMM. The diagnostic tool that is used in workshops and assembly line is connected to the green bus via a VCI, figure [7].

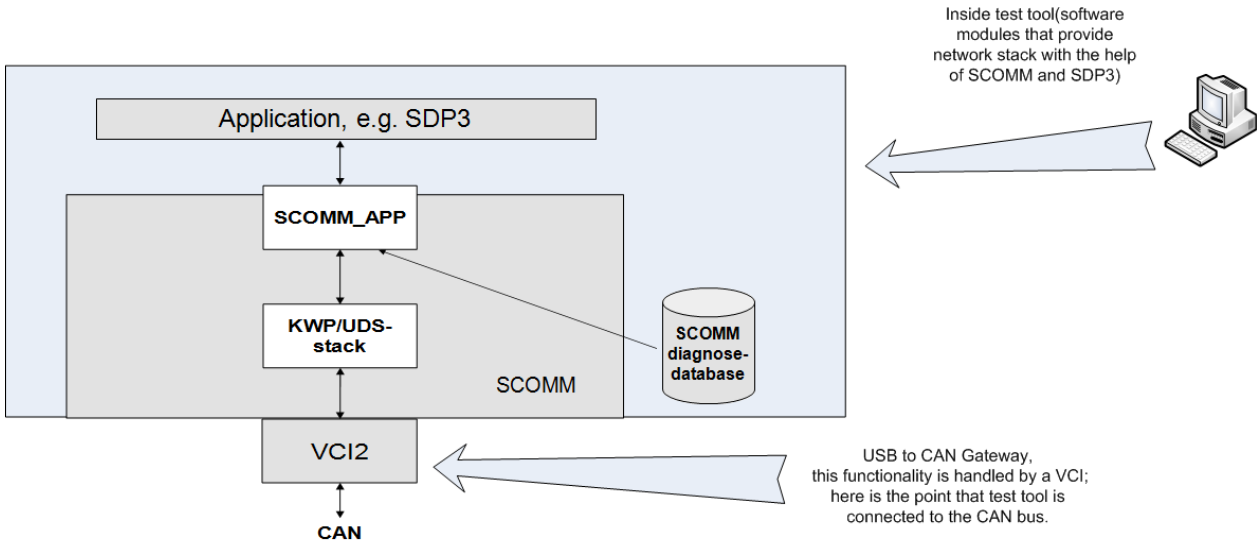

Figure 7. Connection Hierarchy

Figure 8 shows the layers that UDS and DoCAN are applicable to:

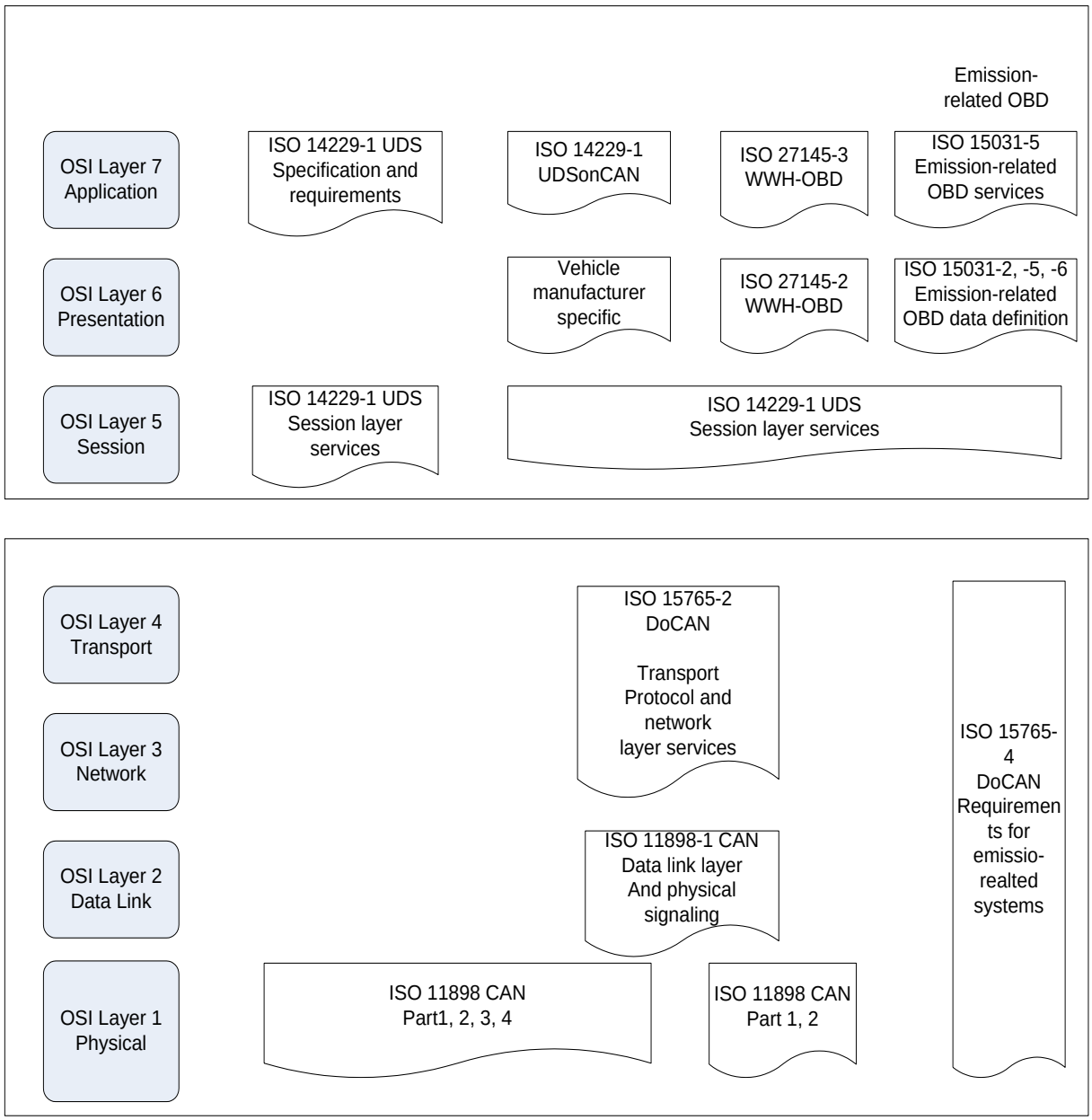

Figure 8. Diagnostic Protocols and related network Layer

Below I will go through use cases of the currently used protocol for on-board diagnostics:

DoCAN use cases:

1. Vehicle inspection and repair: this use case requires that the data needed for vehicle inspection and maintenance be available via the CAN bus from the vehicle without any connection establishment or security negotiations.

2. Vehicle/ECU software reprogramming: this use case needs the in-vehicle CAN network to have the possibility to do update software/calibration of ECUs with connection establishment and security negotiation.

3. Vehicle/ECU assembly line inspection and repair: to fulfil this use case all data that are needed for vehicle/ECU assembly must be available through the CAN network from the vehicle without any connection establishment and security negotiation.

A short description of the functionality of each layer is revealed in the rest of the chapter.

Network layer:

This layer is in charge of segmentation, reassembly and transmission with flow control. If a message does not fit into a single CAN frame then it is sent into multiple parts according to the size of the CAN frame. The services provided by this layer to the above layer are either communication services or protocol parameter setting services. The communication services enable the transfer of up to 4095 bytes.

These two main categories of services have got the same format and they can be a service request, a service indication or a service confirmation. The service request is used by the higher layer to pass the data to the network layer.

The service indication is used by the network layer to pass the status and the received data from lower layer or the higher layer.

The service confirmation is also used by the network layer to pass only the status information to the higher layer.

The purpose of the flow control frame is to synchronise the sender with receiver capabilities.

There are two target address types: physical and functional. The physical addressing is 1 to 1 communication whereas functional addressing is 1 to n communication.

Each communicating peer is supposed to function in a way so that performance requirements are answered. Error detection and timeout are some of the cases that affect the performance. On the establishment of a communication path between peer protocol entities, a single set of network layer timing parameters is assigned to this path to supervise the healthy functionality of the communication session.

All these services are independent of any implemented application. Figure 9 shows how the network layer handles messages that are larger than a CAN frame size [40].

Transport layer:

This layer is responsible for delivering of messages up to 4095 bytes and reporting the failure or the completion of transmission and reception. Messages larger than 4095 bytes are fragmented in this layer.

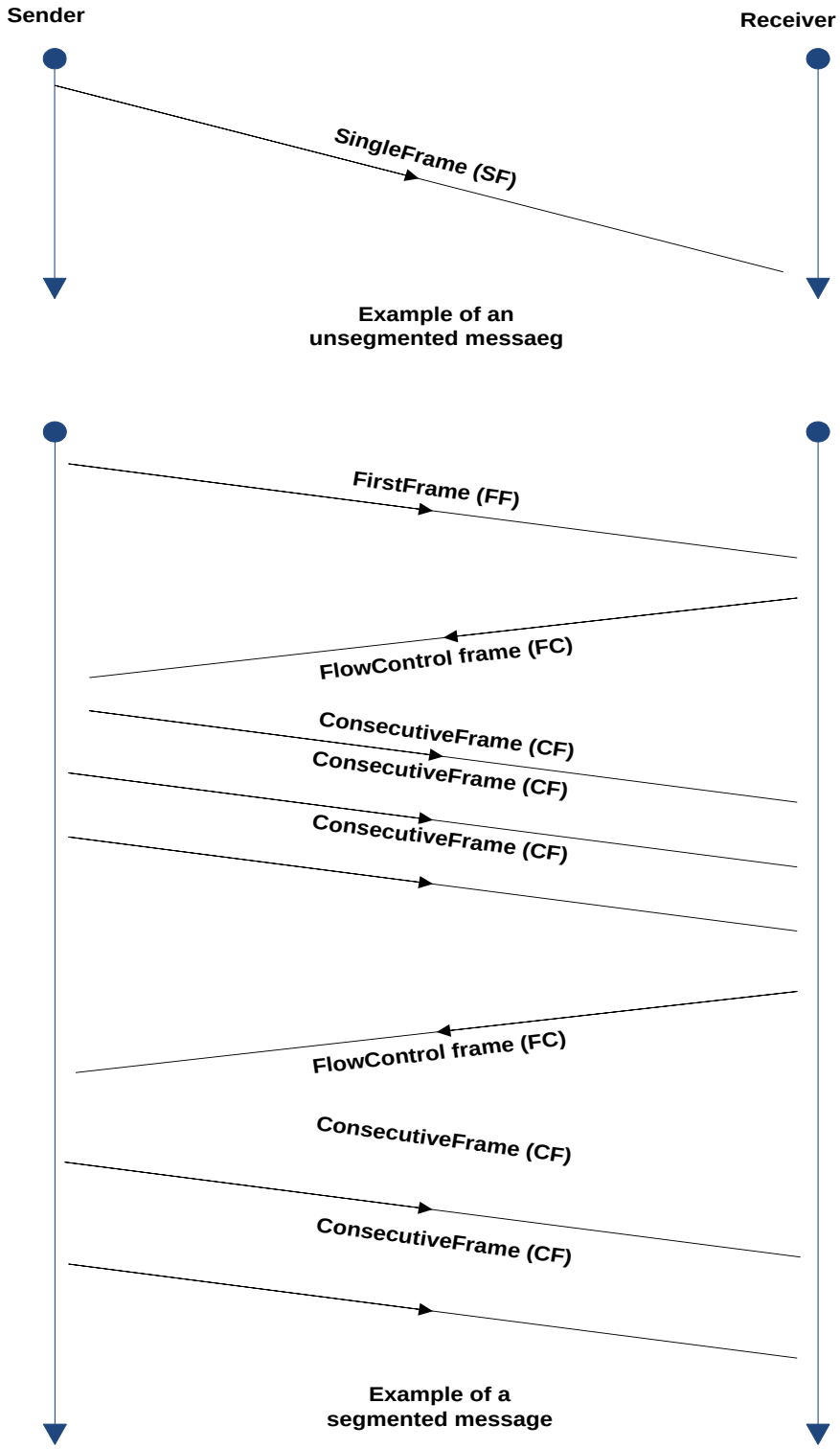

Figure 9. Segmented and Unsegmented Message

Figure 10 illustrates how a message is passed between layers from transmitter to receiver [41]:

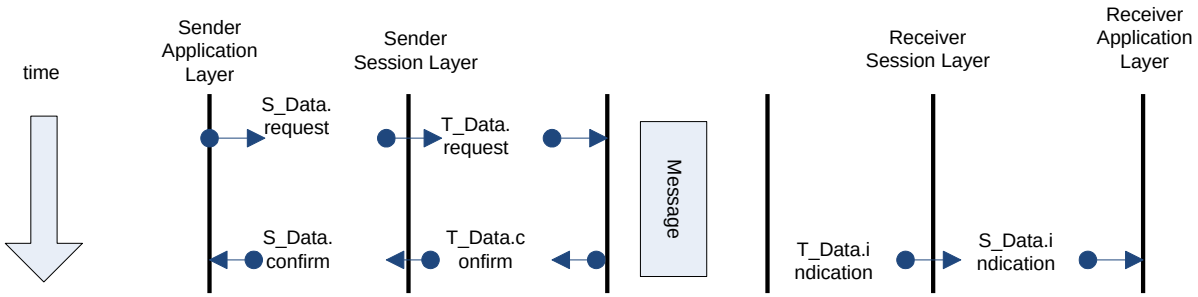

Figure 10. Message Flow between Layers

The specification for application layer, presentation layer and session layer is included in UDS specification [36]. This protocol defines a set of services for testing, monitoring, inspection and programming.

Requirement for time handling in the application and session layer are included in this specification. Usually timing values defined for worst case scenario are affected by:

- ✓ Number of gateways in the path connection source and target nodes,
- ✓ Baud rate,
- ✓ Bus utilization,
- ✓ Processing time and driver implementation(polling or interrupt)

3.2 Diagnostics over Internet Protocol (DoIP)

A review of previous works and current works shows that Ethernet deployment for real-time control data and infotainment, which are bandwidth-hungry, is getting more popular. Maybe no one would think of Ethernet in the vehicle industry, since it has a high bandwidth which was not required in the early stage of networking in the vehicle industry.

In the beginning there were just some ECUs that were transmitting control data and parameters in between. Growth of technology was along with bandwidth-hungry applications.

There are two facts that are good to remember before moving further:

- ✓ In the diagnostic domain we are not dealing with requests that need bounded delay.
- ✓ Treating a non-real-time system as a real time system increases the network load and limits system resources.

Following are some of the characteristics that make Ethernet a candidate for the vehicle industry:

- ✓ Cost effectiveness,
- ✓ Quick start up,
- ✓ Safety requirements,
- ✓ Long maintainability,
- ✓ Scalability,
- ✓ It is being used in industrial environment – critical environment ,
- ✓ High EMC resistance,
- ✓ Low weight,
- ✓ Not spacious,
- ✓ Power efficiency.

Diagnostics over IP (DoIP) is a new protocol that has been introduced in the vehicle industry with a broader range of use cases and ease for users.

The intention of this protocol is separating the in-vehicle network technology from the external test devices and helping the manufacturers to make use of the existing communication standards that are compatible with the legislated diagnostic communication protocols. It also introduces a standardized vehicle interface that can be adapted to new physical and data link layers, regardless of being wired or wireless.

Table 7 demonstrates the layers of the OSI model that the DoIP protocol is involved in:

ISO 13400 is mainly defined for diagnostic systems, but can be used for other services. A list of the use cases that this protocol covers is described below:

- ✓ Retrieving a predefined amount of data from a vehicle without any need for connection establishment or security,
- ✓ Quick availability of data that are needed for inspection, maintenance and repair over the IP network without any need for connection establishment or negotiation,
- ✓ Programming or updating ECU software with connection establishment and security negotiation,
- ✓ Quick availability of all the data that is needed for inspection, maintenance and repair in the assembly line without any need for connection establishment and security negotiation,
- ✓ Retrieval of non-diagnostic data, such as address book, e-mail from infotainment components and transferring it to the vehicle or vice versa.

In order to make the services reusable like application layer services and transport layer services and hide the protocol from the service user, DoIP distinguishes between the provided services by a layer, from the ones provided by the layer above and also it distinguishes the used protocol for communicating between two peer entities. This approach makes the services reusable for other networks than IP, like CAN.

Table 7. DoIP and UDS

<table>
<tr><th>Applicability</th><th>OSI 7 Layers</th><th>Vehicle Manufacturer Enhanced Diagnostics</th><th>WWH-OBD Document Reference</th></tr>
<tr><td rowspan="7">Seven layers according to ISO 7498-1 and ISO/IEC 10731</td><td>Application (layer 7)</td><td>ISO 14229-5/ISO 14229-1</td><td>ISO 27145-3/ISO 14229-1</td></tr>
<tr><td>Presentation (layer 6)</td><td>Vehicle manufacturer specific</td><td>ISO 27145-2, SAE J1930-DA, SAE J1979-DA, SAE J2012-DA, SAE J1939, SAE J1939-73</td></tr>
<tr><td>Session (layer 5)</td><td>ISO 14229-2</td><td>ISO 14229-2</td></tr>
<tr><td>Transport (layer 4)</td><td rowspan="2">ISO 13400-2</td><td rowspan="2">ISO 13400-2</td></tr>
<tr><td>Network (layer 3)</td></tr>
<tr><td>Data link (layer 2)</td><td rowspan="2">ISO 13400-3</td><td rowspan="2">ISO 13400-3</td></tr>
<tr><td>Physical (layer 1)</td></tr>
</table>

Due to Internet protocol characteristics we can have the following communication scenarios:

- ✓ Direct physical connection between one external test device and one vehicle,
- ✓ Connection over a network between one external test device and one vehicle,
- ✓ Connection over a network between one external test device and multiple vehicles, and
- ✓ Connection over a network between multiple test devices or multiple test applications on a single physical device and one vehicle.

There are parameters that need to be considered at the time of applying use cases and application layer protocols (timing is one important factor) to a specific network configuration, like dropped packets and errors, delay and jitter, out of order delivery, transmission rate, IP address assignment and vehicle discovery. Table 8, derived from ISO13400-1, gives a qualitative overview of these characteristics over typical network topologies [42].

Table 8. DoIP and Communication Scenarios

Characteristics		Direct connection	Switched network	Different sub-networks over private network	Different sub-networks over public network
Transmission rate		Predictable	Predictable(require higher efforts)	Predictable(require higher efforts)	Unpredictable
		Dedicated bandwidth available	Average shared bandwidth guaranteed by network design	Average shared bandwidth guaranteed by network design	No guaranteed bandwidth
		e.g. 100 MBit/s for 100BaseTx Ethernet	Individual bandwidth depends on overall network traffic	Individual bandwidth depends on overall network traffic	
Dropped packets and errors		Extremely low probability	Low probability	Average probability	High probability
Delay and jitter		No delay	Low delay	Average delay	Varying delay (low … high)
		No jitter	Low jitter	Average jitter	High jitter
Out-of-order delivery		No	No	Possible	Possible
IP address	Link-local Auto IP	Supported	Supported	Unusual	Unusual
				Would require special router configuration(e.g. NAT)	Would require special router configuration(e.g. NAT)
	Private IP address	Supported	Supported	Supported	Unusual
					Would require special router configuration (e.g. NAT)

	Public IP address	Supported	Supported	Supported	Supported
Vehicle discovery		Link-local Broadcast	Link-local Broadcast	Subnet-specific broadcast(requires knowledge about network topology)	Subnet-specific broadcast(requires knowledge about network topology)
		Unicast (requires knowledge of vehicle's IP address)	Unicast (requires knowledge of vehicle's IP address)	Unicast (requires knowledge of vehicle's IP address)	Unicast (requires knowledge of vehicle's IP address)

The physical layer of a DoIP stack is based on Ethernet and the network layer is consistent with IEEE 802.3. The defined limitation by a MAC layer on a maximum transport unit is about 1500 bytes. It is worth mentioning that higher layers are responsible for fragmentation. The entities can either work on IPv4 or IPv6, i.e., all entities shall implement the same protocol.

Figure 11 shows the protocols and their place in the OSI standard that need to be implemented in each DoIP entity:

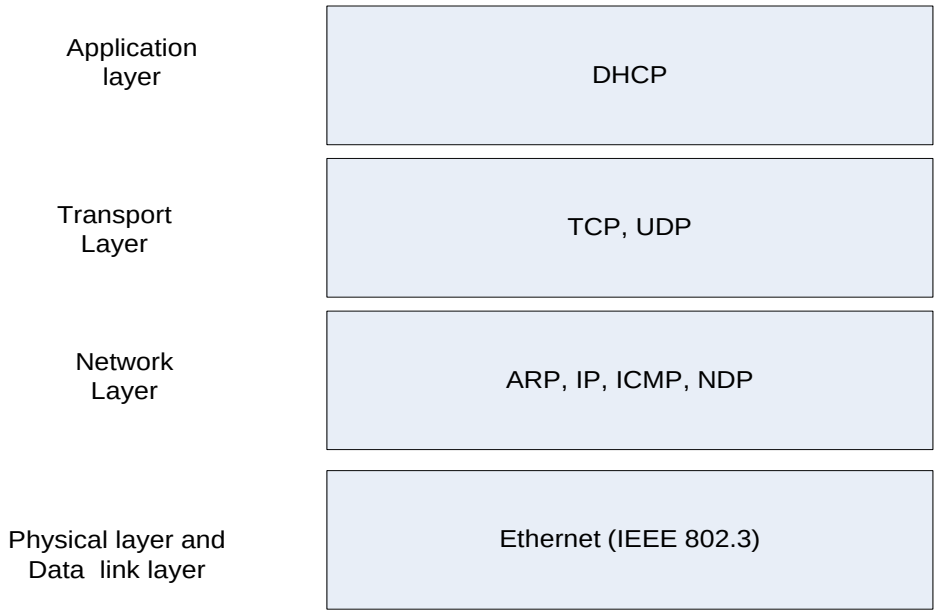

Figure 11. IP Stack

Use of transmission control protocol (TCP) guarantees reliable and in order delivery but it puts more overload on the network, which is why in some of the DoIP use cases, user datagram protocol (UDP) is suggested.

The communication is done by a so called socket mechanism and the reserved port for communication is 13400 and 13401. The data transmission order required by this protocol is based on big-endian network byte order.

The DoIP payload is grouped into three categories: Node management, vehicle information and diagnostics.

Each DoIP message has got a header which specifies the version of the DoIP packet, type and length of the payload. Depending on transport protocol, UDP or TCP, the number of messages that can be sent on each datagram is different. In UDP only one DoIP message can be sent in each datagram and it starts with a generic header and in TCP several messages can be sent that are separated by the generic header.

All defined scenarios are done via socket connection. Sockets provide a logical connection between test tool and DoIP entities. Each socket is defined by a source address and a socket handle. The socket handle represents source address, destination address, port and transport protocol. These properties make one socket distinguishable from another socket.

Below is a brief description and illustration on how a DoIP session is handled:

After the activation signal triggered by the test tool, the DoIP gateway will start the vehicle identification announcement session and then the connection is initiated from the test equipment to the DoIP entity inside the vehicle by connecting a socket to the listening port on the entity. So the DoIP entity is responsible to provide sufficient resources for incoming connections. After the socket is opened, required timers; inactivity timer and generic timer, are assigned and the routing procedure will start. In the routing session, the source address will be associated with the socket and after successful routing the diagnostic session will start which is followed by requests and responses from/to ECUs. Being done with requests and responses, if the connection is not needed any more the test tool is to close it through TCP/IP protocol mechanisms to release the resources. If the connection closure fails for any reason, it will be done after a timeout signal by the inactivity timer assigned to that socket [43]. Figure 12 illustrates a DoIP process and figure 13 provides a general view of the network architecture in DoIP.

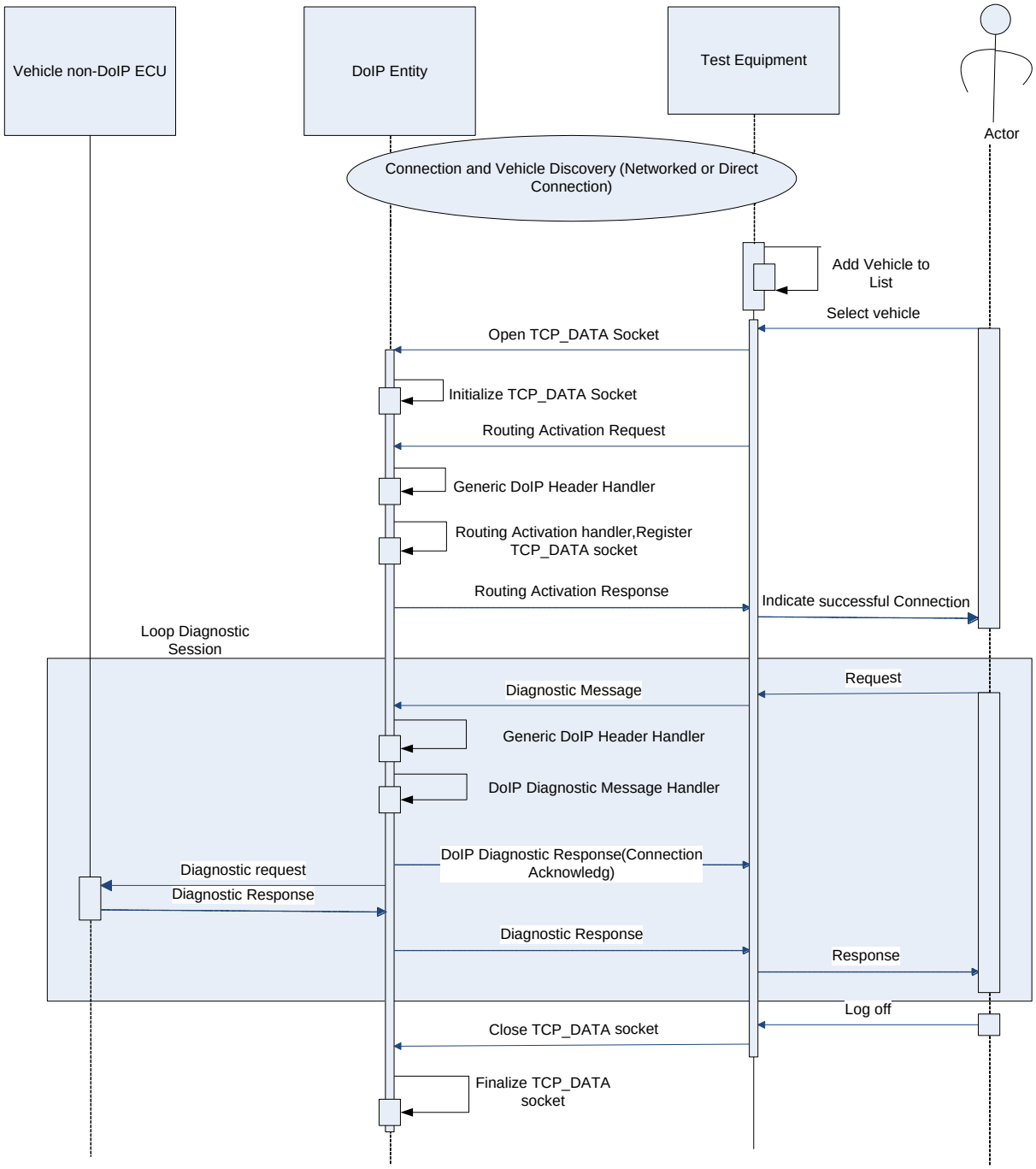

Figure 12. A DoIP Session-Sequence Diagram

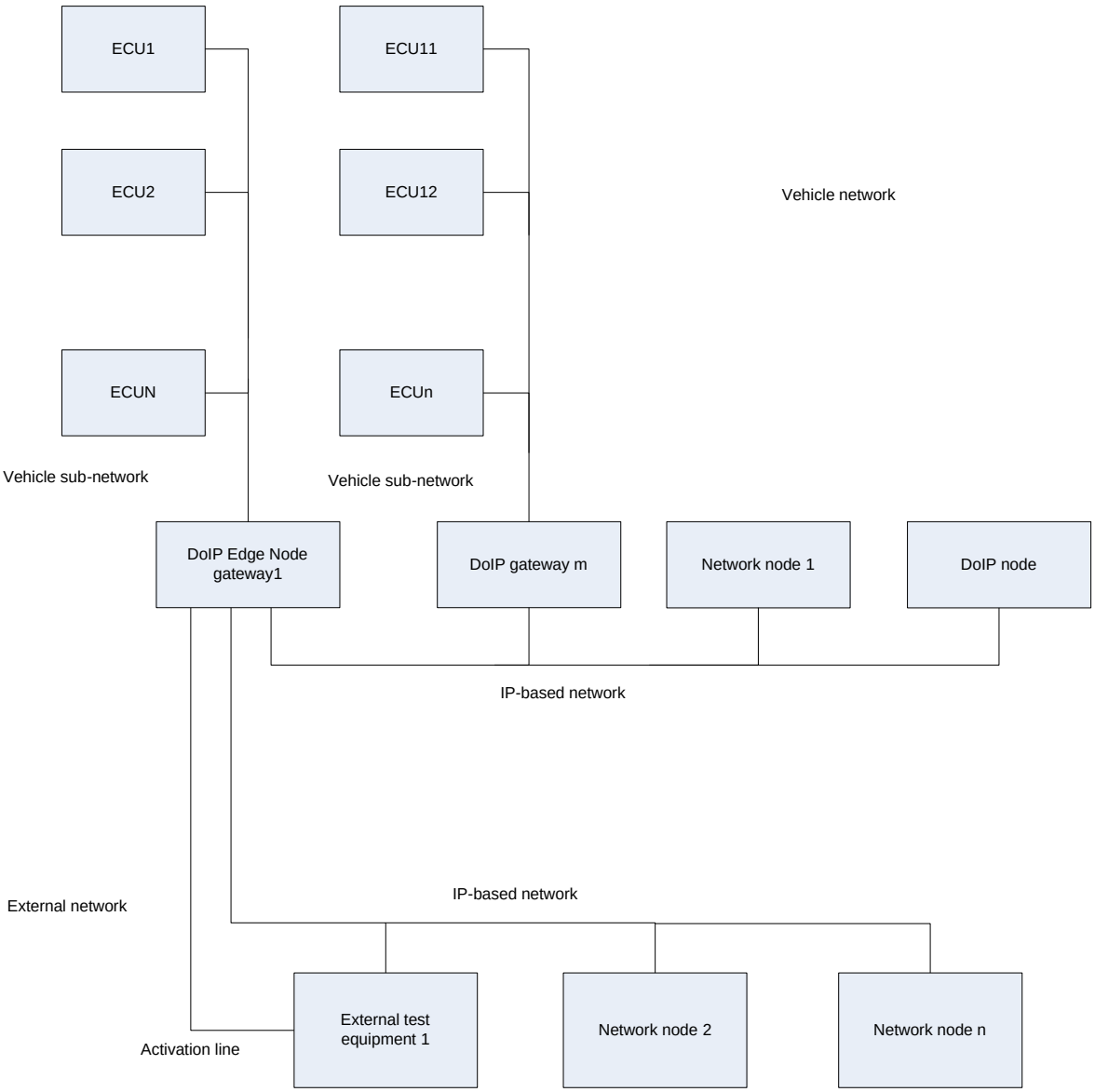

Figure 13. Network View

4 Design and Implementation

This chapter will introduce two models and provide detailed steps regarding implementation.

4.1 Challenges

Implementing a new protocol on a system which is currently working and is under development has got its own type of challenges. There are some specific requirements that are needed by UDS and DoIP protocol so that the system is consistent with the specification, such as size of source address and target address, length of payload, existence of both KWP2000 compliant and UDS compliant ECUs on the same vehicle (DoIP specification only supports UDS protocol as its higher layer protocol) and the possibility for the DoIP gateway to have a list of available ECUs on the vehicle. These requirements are not consistent with what the manufacturer has defined in its use cases and needs discussion to be handled. Ignoring these requirements for the current work is not going to affect the general concept of the thesis work.

4.2 Suggested Approaches towards the Solution

In this section I mention two approaches that can be taken to implement the solution and in the next section I introduce a two level model for each of the approaches. My suggested approaches are inline with manufacturer strategy and satisfy the limitations described in section 1.2, as would be seen in the practical work. Following are the approaches that can be taken to deploy the proposed solution in section 3.2:

1. Adding the DoIP gateway to the green bus, only as a diagnostic gateway. In this method the introduction of the gateway is just for diagnostic purposes. We call this method “first step” for later reference.

2. Deploying the DoIP gateway connected to a full-duplex switch to reduce the traffic on the green bus by moving those ECUs that are functioning on the green bus to the Ethernet network by connecting them to the full-duplex switch. This method can be built upon the first step which was described in the first approach that provides a gradual development in the process and is more consistent with Scania’s strategy and future plan of “Ethernet for In-Vehicle Network”. We call this method “second step” for later reference.

4.3 Proposed Model

There are a lot of models that can be deployed on a general approach but the need to follow some limitation makes it a bit difficult to determine the proper model. My model is in two levels and differs a bit for each of the approaches that were mentioned in section 4.2. In fact models related to the second step are complementary models for the first step.

4.3.1 First Level Model consistent with First Step

Figure 14 illustrates the first level model of first step. The reason that we place the gateway on the green bus is due to the characteristics of diagnostic messages. Diagnostics traffic does not have high priority and that is the case with green bus traffic. Moreover, the current diagnostics which is on CAN is being done on the green bus, so installing the gateway on the green bus is more consistent with the current state of the system. COO isolates the traffic of three CAN buses.

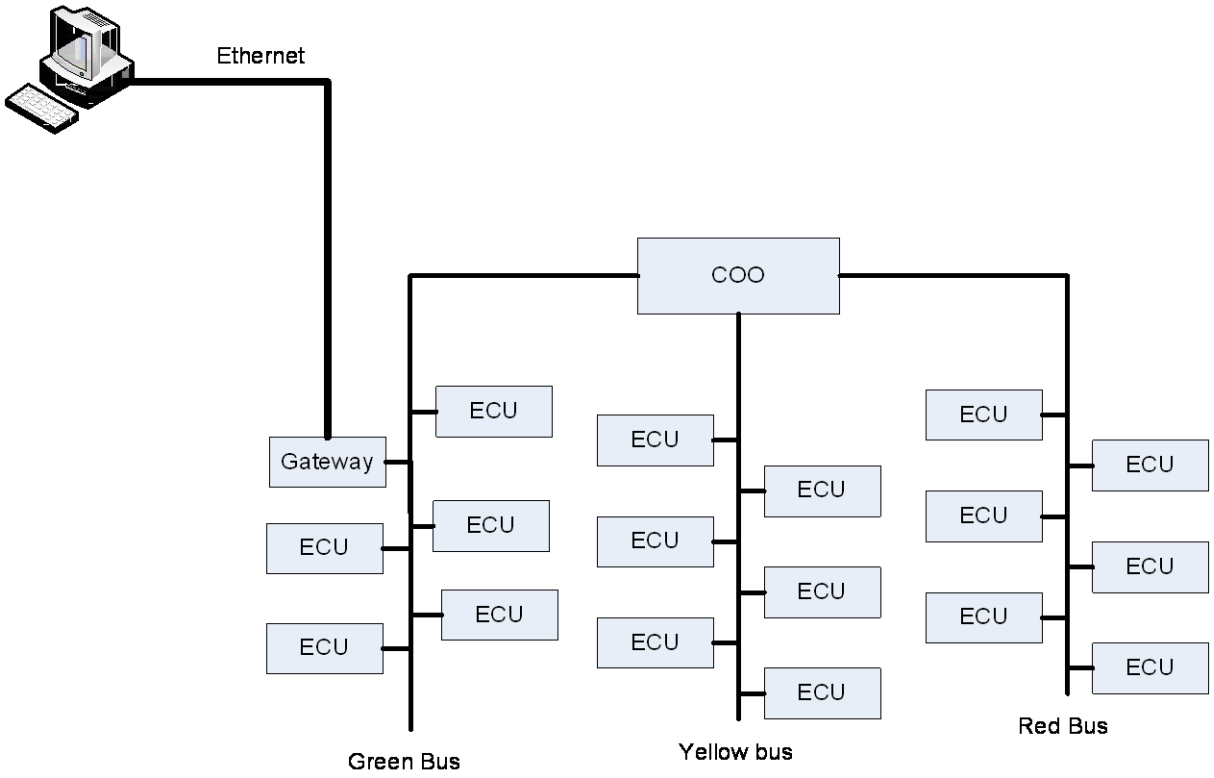

Figure 14. First Level-First Step Model

4.3.2 First Level model Consistent with Second Step

As can be seen in figure 15, I have added a full-duplex switch and some ECUs to the model compared to the one in the first step. The reason behind moving some ECUs to the Ethernet network is reducing traffic on the green bus. It is recommended that those ECUs that have more interaction together, need higher speed and have a high volume of data, be transferred to the Ethernet network. The aim of this model is to pave the way for a better utilization of Ethernet advantages.

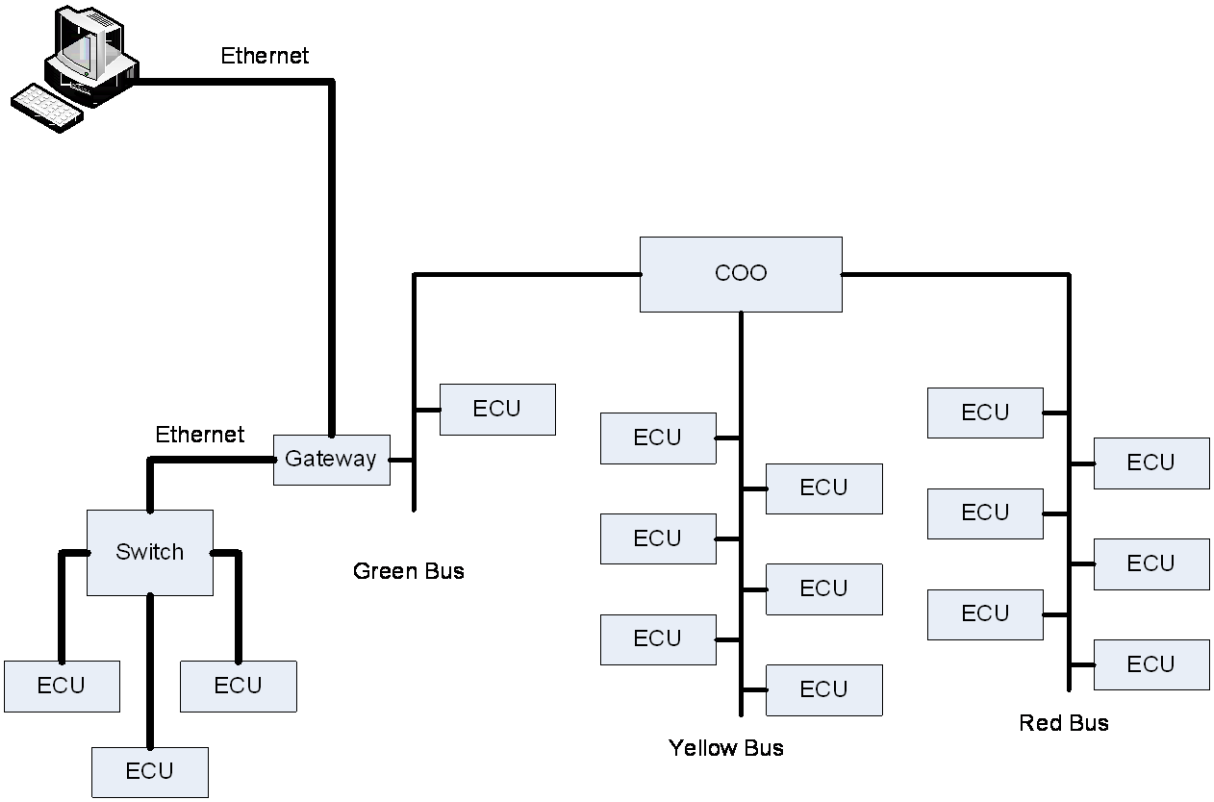

Figure 15. First Level-Second Step Model

4.3.3 Second Level Model Consistent with First Step

In this section I will introduce a model called second level model which gives a better view of system elements and the requirements for implementation. This can be viewed in figure 16.

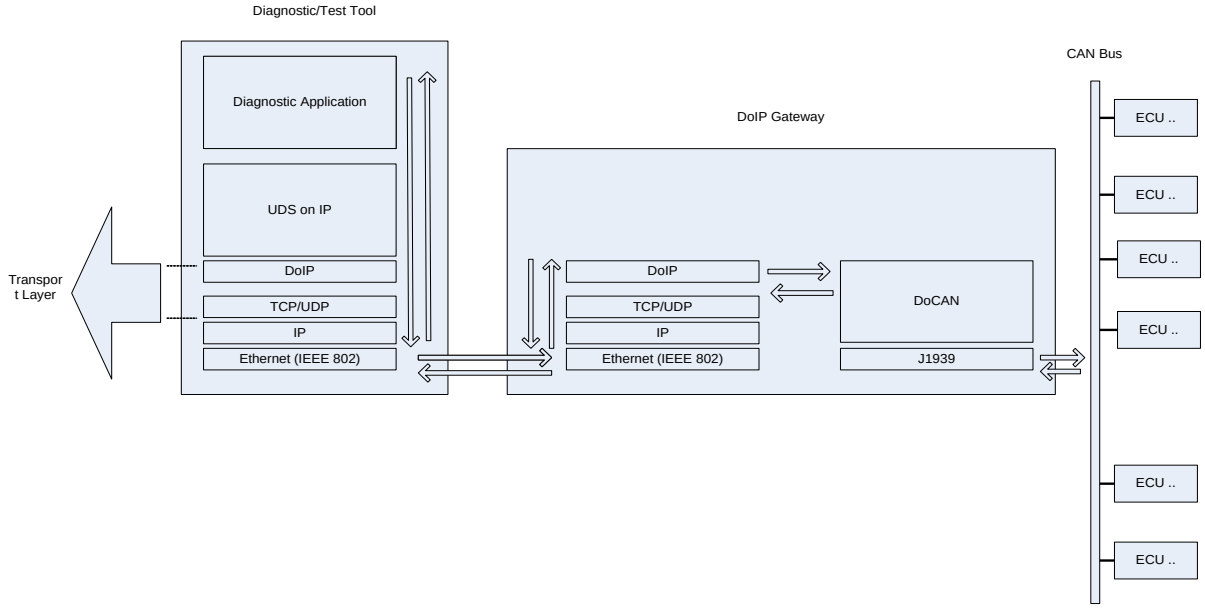

Figure 16. Second Level-First Step Model

Applications regarding diagnostics are of no real-time requirements. Knowing this fact helps the designer to avoid the resource handling and synchronization that usually put lots of load on the system. In the gateway, the network stacks are provided by several software modules which are independent of each other. I make use of this model in my prototype.

4.3.4 Second Level Model Consistent with Second Step

Like the first step, I introduce a model called second level model which has got a general perspective, compared to only-diagnostic scenario as was the case with the first step. This is illustrated in figure 17.

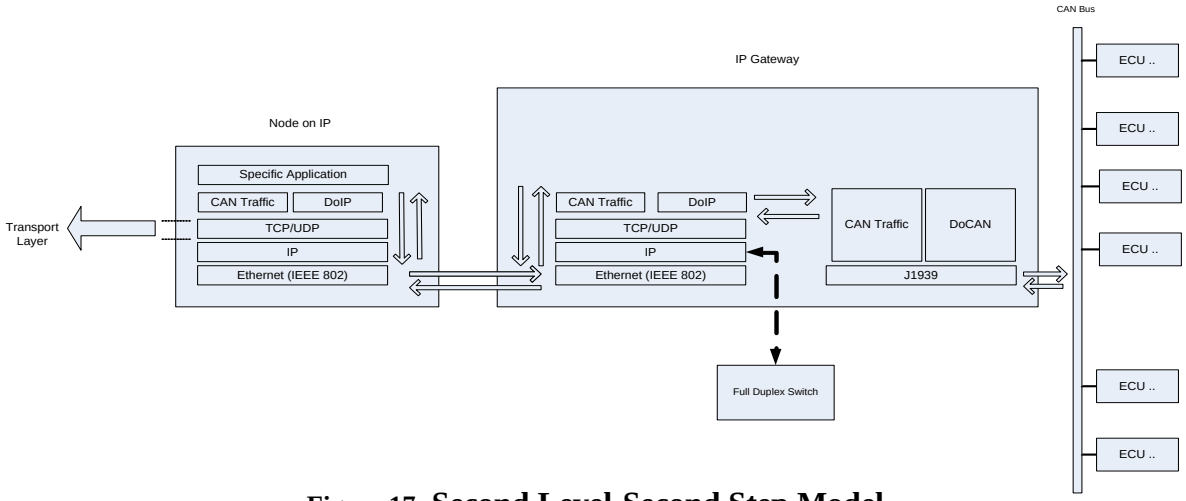

Figure 17. Second Level-Second Step Model

Ethernet is mainly developed for applications that are in need of high bandwidth, whereas CAN has been developed for real-time applications and bounded delay and jitter are the main objectives. These differences make the deployment of the second step a bit challenging.

In order to provide the real-time requirement of the second step, several solutions have been proposed [19] [44]. We propose our solution inspired from [19] and analyse it theoretically and suggest further investigation and analysis, either by simulation or in practice as a future work.

4.3.5 Some analysis and Suggestion on Second Level Model of Second Step

There are some main differences between CAN and Ethernet which originate from their characteristics and their main objective. Due to these noticeable differences a proper strategy for utilising Ethernet in this domain is of great importance. Before elaborating on the analytical part I will list some of these differences:

- ✓ Difference in available bandwidth: 250kbs (Scania specific) for CAN and 100 (or 1000) Mbs for Ethernet,
- ✓ Difference in addressing: identifier for CAN and MAC addresses for Ethernet nodes,
- ✓ Difference in payload size: 0-8 bytes for CAN and 46-1500 bytes for Ethernet
- ✓ Difference in handling collision management: CSAM/CR for CAN and CSMA/CD for Ethernet
- ✓ Difference in determinism: MAC layer type in CAN makes it deterministic and non destructive whereas in Ethernet makes it non-deterministic and destructive.

At the time of reception of a CAN message the identifier will be checked bit by bit and the associated IP address will be set as the target address and the data part as the payload. In order to distinguish the CAN traffic from diagnostic requests, the protocol type will be set as “CAN traffic”. In mapping CAN addresses to IP addresses each CAN address is mapped to an IP address of the nodes connected to the switch.

4.3.6 How to fulfil the real-time requirements

Here we will consider two steps to meet the hard-real time requirements. First, each received CAN message will be fed to a separate buffer associated to its priority and identifier and the data will be inserted into the buffer in a FIFO method. Second, a timer will be assigned to each buffer with a timeout value of a fraction of the relative deadline of the already received request. If no other message with the same priority and identifier is received prior to the timeout, the buffer will be sent as payload and will be cleared for the reception of new requests. In this way I will guarantee that no deadline is going to be missed. On the other hand, by using a full-duplex switch, a bounded jitter for periodic messages will be guaranteed. The reason behind introducing the timer and sending more than one request per Ethernet packet, on one hand is saving more bandwidth due to the difference in CAN and Ethernet payload and on the other hand, reducing the traffic on the IP network which is a shared resource.

Once there is a request from the IP network to the CAN bus, the data and identifier will be constructed in the source IP node and sent as an IP packet to the gateway. Each received packet on the gateway will be queued in an output buffer associated with its identifier, and if the size of the payload is less than 8 bytes, the message will be sent on the bus immediately and if it is more than 8 bytes it will be sent after fragmentation.

4.4 Detail on Prototype

The practical work has been done close to the proposed models for the first step, as introduced in section 4.3.1 and 4.3.3. The designed gateway is connected to the yellow bus of the vehicle with some ECUs in a way that the general concept that is needed for test validation is fulfilled. The software module that is in charge of DoIP has been developed during the work and gets connected to other software modules that are already implemented to carry the diagnostic requests. Since some of the modules have been already implemented and were being used and due to time limitation, it was not possible to design the whole system as the proposed model states, but the requirements are still fulfilled. In the following sections, I elaborate more on the work.

4.4.1 Elements of the System

ECUs, gateway, CAN bus, Ethernet Connection and a test tool are the main elements of the system. I used a PC as the test tool and its operating system is Windows 7. The gateway is an ECU that is used for fleet management. In fact, fleet management handles remote diagnostics and ITS functions where the controller in charge is on the yellow bus. Linux/Debain is used as the operating system and Freescale MPC series is selected as the hardware. The programming language used in both gateway and test tool is C++. Figure 18 illustrates the connection between elements.

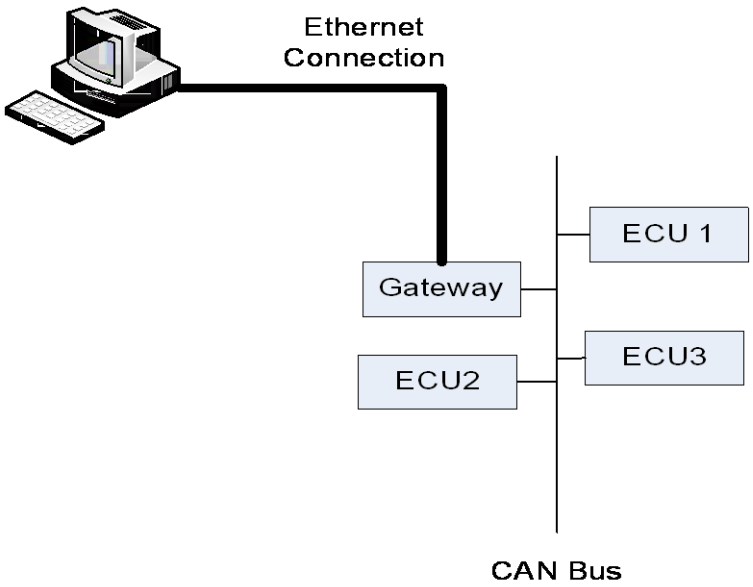

Figure 18.Connection Between Elements

4.4.2 Development Environment

Netbeans IDE version 7.1 is used as the development environment for the server application on gateway. The developed application is cross compiled on the gateway operating system which is Linux. The cross compilation is done on the Linux kernel which is hosted by Oracle VM VirtualBox since the cross compilation which is offered by Netbeans is not bug free. After cross compilation the binary data is secure copied to the gateway via an Ethernet connection [49].

Client application on the test tool is developed on Microsoft visual studio 2010. Figure 19, figure 20 and figure 21 are related to development environment of the test tool and gateway.

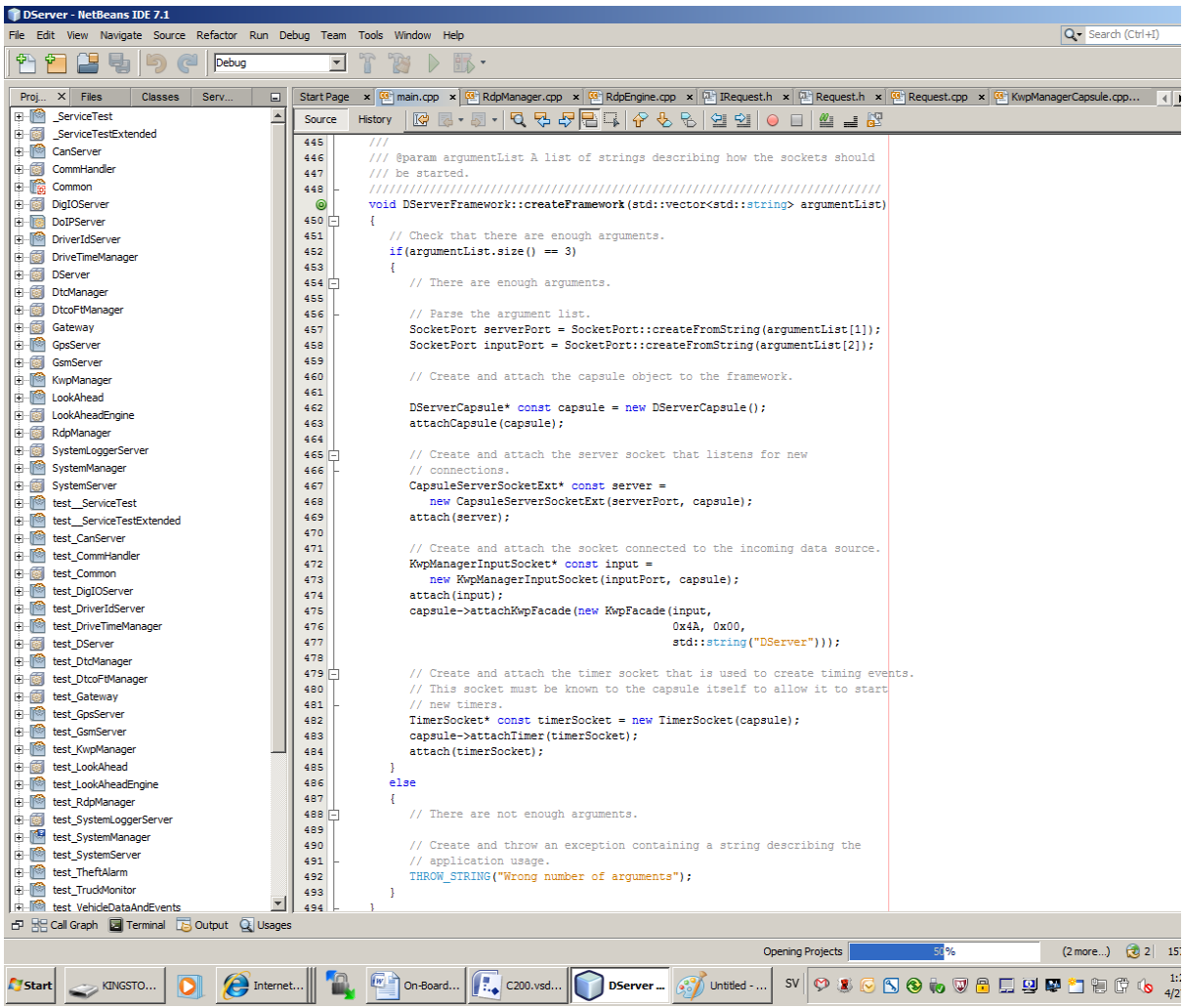

Figure 19. Netbeans IDE

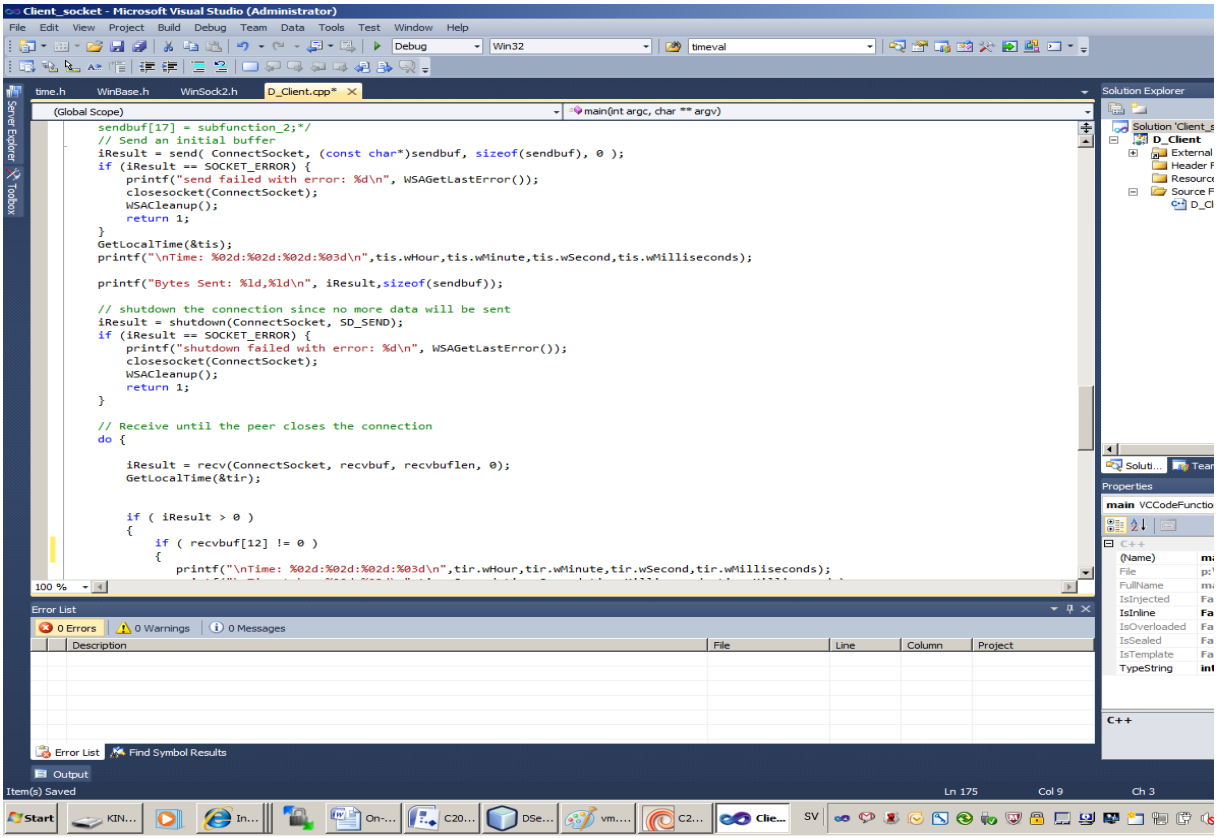

Figure 21. Microsoft VS 2010

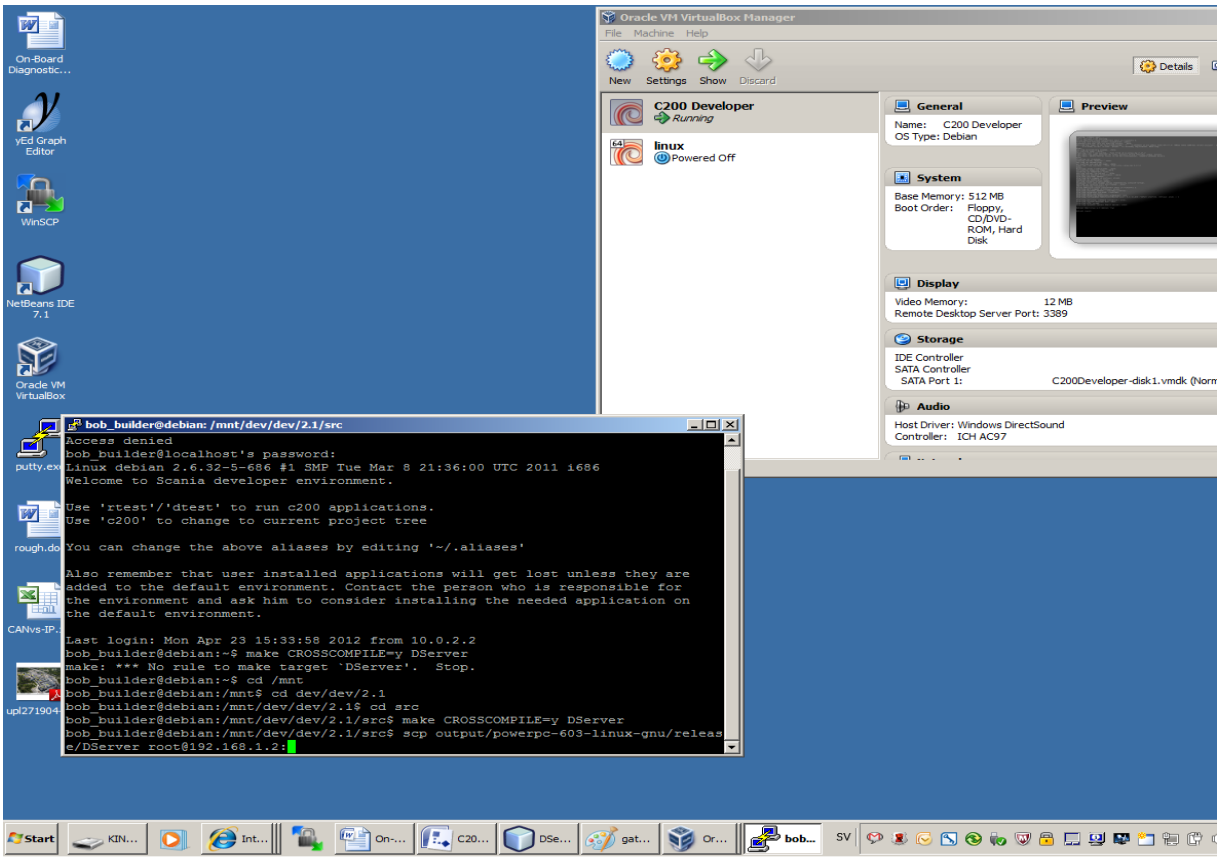

Figure 20. Virtual Box

4.4.3 Design

In the gateway, the network stack is provided with several software modules which are independent of each other. Some of these modules have been already coded and some not. Each module is in a form of application that is encapsulated into a framework and the framework can get access to the hardware and memory resources that are managed by the operating system (Fig. 22).

As illustrated in figure 22, “Timer Socket” takes care of the timing task by assigning a timer to each request and checking the value for a probable timeout. Connection to other software modules which provides some of the network stack is done by “Input Socket”. “Server Socket” and “Client Socket” to take care of incoming connections from the test tool. The logic of each application and specifically its task is coded in “Capsule”. In my work, “Capsule” validates the data sent by the test tool and after deriving the payload and request type, will forward it to the next application which is called DoCAN.

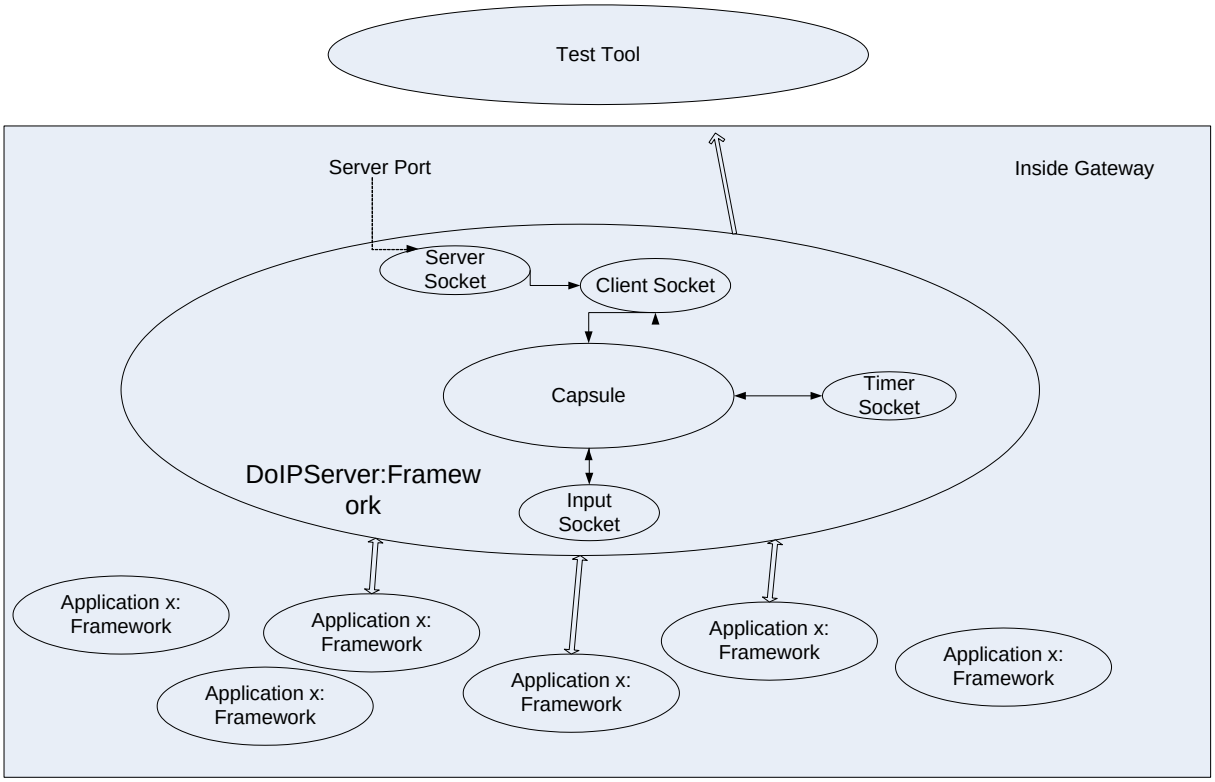

Figure 22. Software Architecture in Gateway

In the DoIP gateway, after receiving the trigger signal from the test tool and executing the vehicle announcement and routing session, the diagnostic server (DoIP Server) will start listening for any incoming request. DoIP Server is the software modules that we have mainly developed. The logic of this module is coded in capsule which is implemented in the capsule. The logic is as the specification (ISO 13400) has specified. The signal flow between software modules is illustrated in figure 23. This figure just shows the software modules that are needed for a DoIP session regardless of software modules and operating systems of the gateway. Figure 24 provides a simplified UML diagram of the designed DoIP gateway.

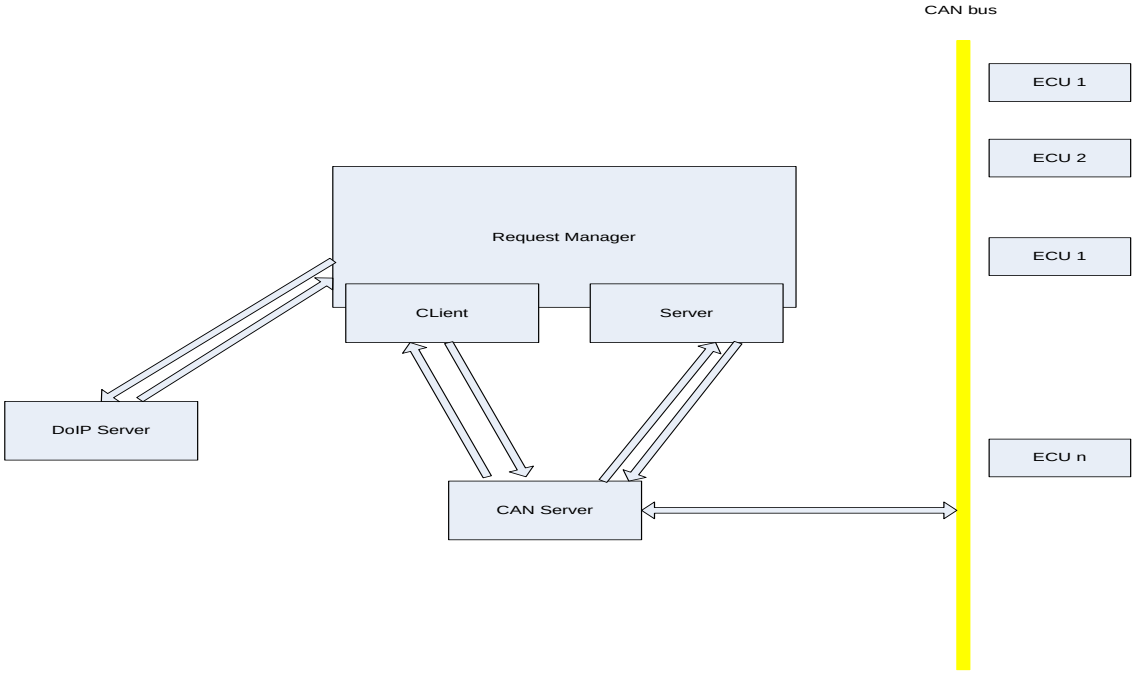

Figure 23. Signal Flow

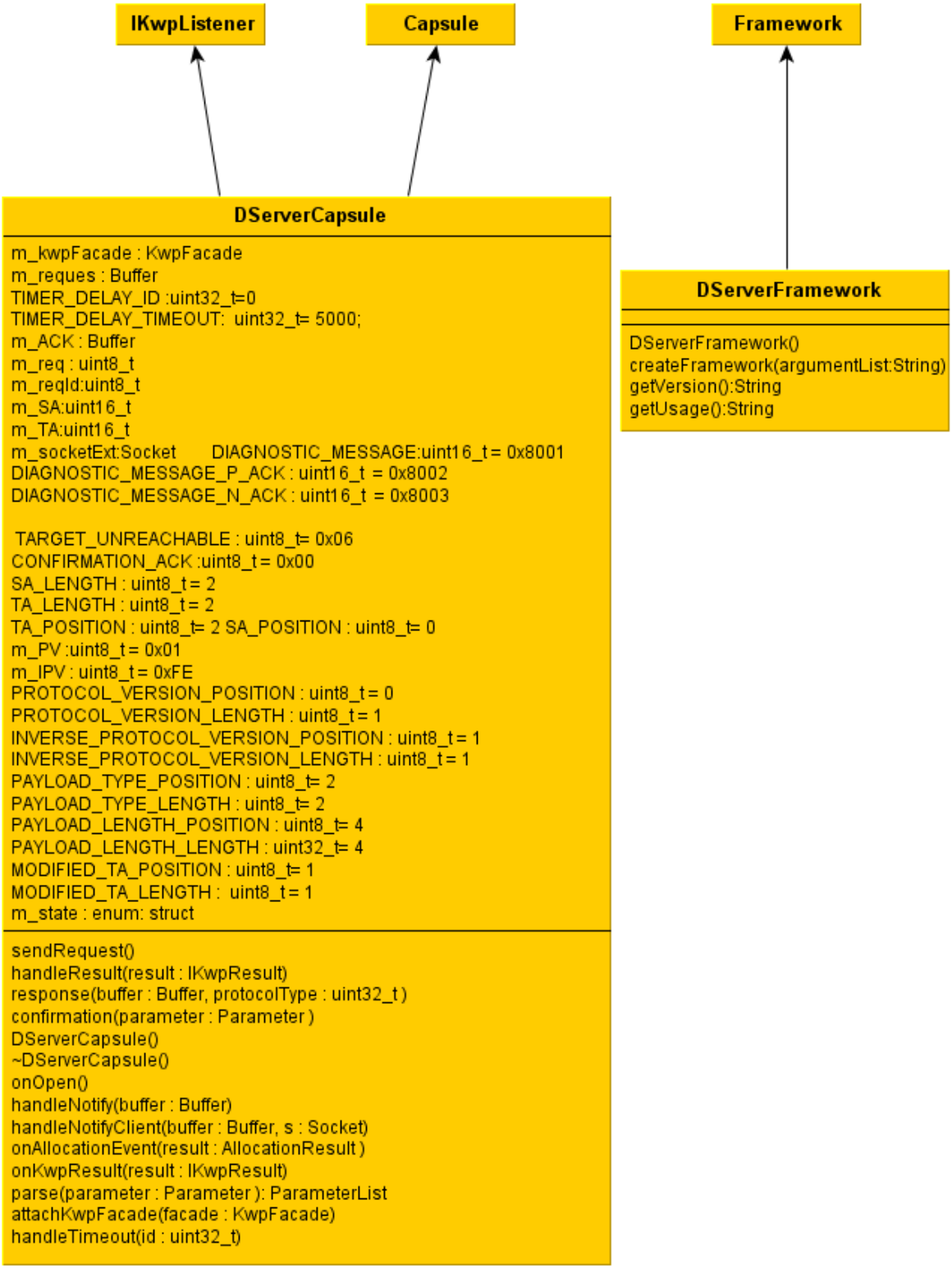

Figure 24. UML Diagram of Server Application on DoIP Gateway

5 Result and Discussion

In the last two chapters I more or less focused on the possibility of DoIP implementation on Scanias' product. Here in this section we intend to investigate the risk of DoIP deployment by running a comparison on response time between DoCAN and DoIP.

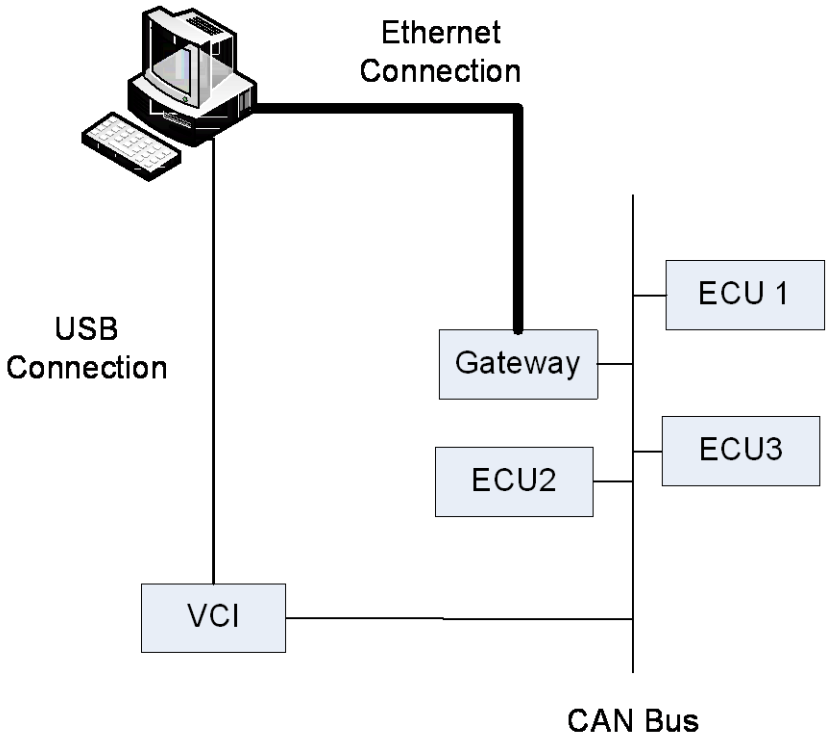

Figure 25. Connection between Test Elements

I have done the test based on the following assumptions:

- ✓ STmin: time interval that the transmitter must wait before sending the next consecutive frame is set to zero in both cases,
- ✓ The derived response time values are average. I ran the test 10 times to make it independent of the delays that are usually imposed temporarily by OS,
- ✓ The transmitting buffer in the IP case is set to 1024 bytes.

The derived response time in DoCAN and DoIP is composed of the following parameters:

$$R\_DoCAN=2*T_{Process\_Test\ tool}+2*T_{Propagation\_USB}+2*T_{Process\_VCI}+2*T_{Propagation\_CAN}+T_{Process\_ECU}$$

$$R\_DoIP=2*T_{Process\_Test\ tool}+2*T_{Propagation\_Ethernet}+2*T_{Process\_DoIP\_Gateway}+2*T_{Propagation\_CAN}+T_{Process\_ECU}$$

$T_{Propagation\_USB}$ and $T_{Propagation\_Ethernet}$ are of the same order and stands for the propagation delay on the USB connection and the Ethernet connection in the mentioned order. The values of these two parameters are almost zero as can be seen in the appendix. $T_{Propagation\_CAN}$ stands for the propagation time over the CAN bus. $T_{Process\_VCI}$ and $T_{Process\_DoIP\_Gateway}$ define the processing time on the VCI and DoIP gateway in the mentioned order. $T_{Process\_ECU}$ describes the time taken by an ECU to process a request which is common in both cases. $T_{Process\_Test\ tool}$ stands for the processing time on the test tool and is negligible since it is the time taken to deliver the request from the transport layer to the physical layer and vice versa.

There are three different types of data in our analysis:

- ✓ Response time of diagnostic requests in DoCAN,
- ✓ Response time of diagnostic requests in DoIP,
- ✓ Response time of an echo request from the test tool to the DoIP Server on the DoIP gateway with a maximum payload size of 1000 bytes.

Test result illustrated in figure 26 shows that diagnostics over CAN is faster compared to IP. There are several reasons that make the result logical:

1. Choice of Operating System, the current work (DoIP gateway) has been done on an ECU which was hosting Linux/Debian which is not real-time. On the other hand, the requests received through the CAN port are treated differently from the requests received from the Ethernet port on the current gateway, which was due to already implemented software architecture which makes it unqualified for a real DoIP gateway. Figure 27 shows a test session where the target node is the DoIP gateway. So it is possible to reduce the processing time in the DoIP gateway by a better choice of OS and a better software architecture, while sticking to the proposed model.
2. In the DoIP scenario, in my work, a request was sent from a PC hosting Windows 7 to the DoIP gateway hosting Linux and from there to the targeted ECU whereas in the DoCAN scenario, the test tool was PC hosting Windows 7 and the request would be sent on the CAN bus via a VCI to the targeted ECU. Being involved with two operating systems, compared to the case in DoCAN which is only one, is another reason for the increased response time. Each operating system introduces some delay to the response time. It should be mentioned that the PC used for the test tool was the same for both the DoCAN and DoIP test session.
3. The confirmation signal is common in both DoCAN and DoIP but in DoCAN it is between the test tool and the target ECU, whereas in DoIP the confirmation is sent by the DoIP gateway to the test tool when the request is delivered to the transport layer of the target ECU. In fact, the existence of the DoIP gateway will cause an increase in response time.

4. The limitation in bandwidth which is imposed by CAN is another reason on overall system performance which causes some kind of bottleneck and does not let having the real throughput of Ethernet. (A general concept)

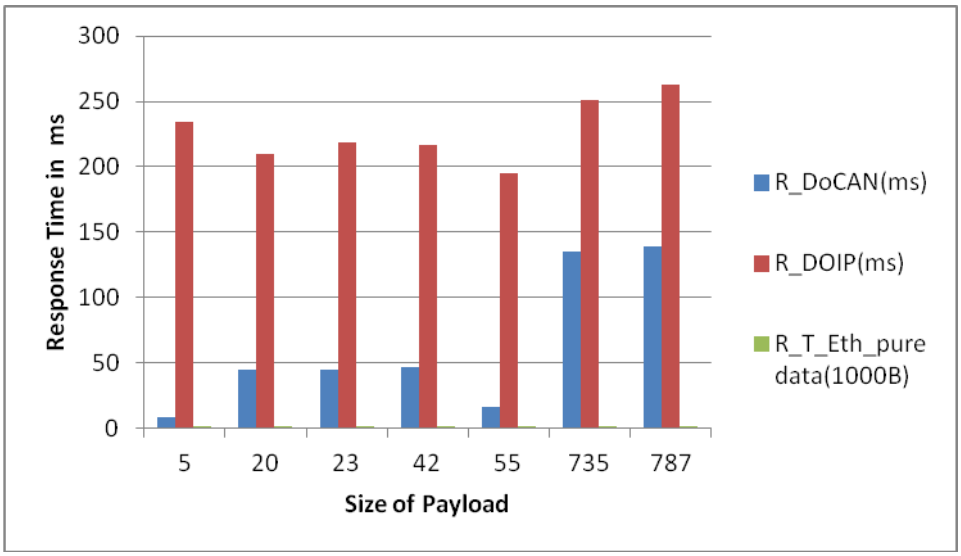

Figure 26. DoCAN vs. DoIP

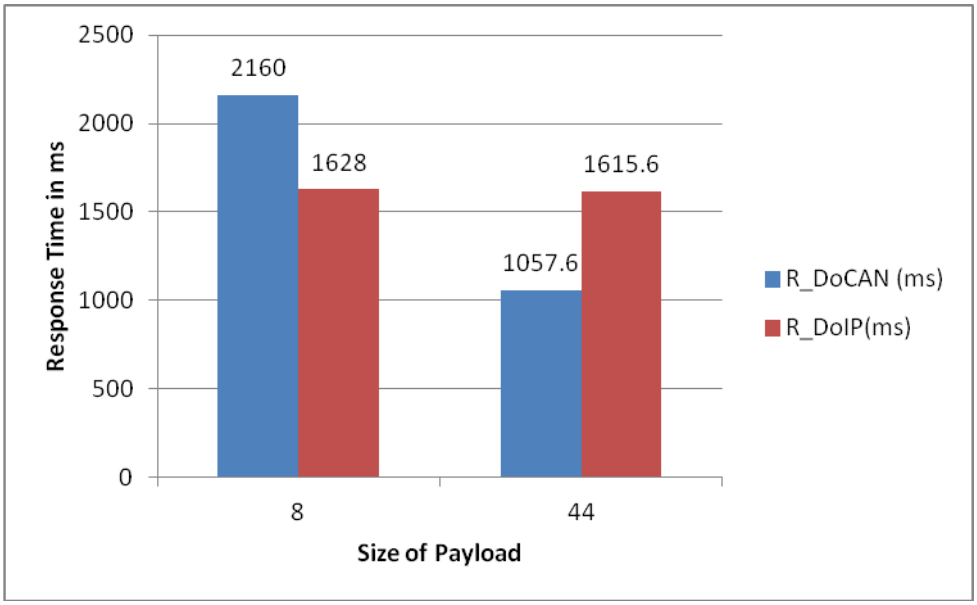

Figure 27. Test Tool to DoIP Gateway

I have mentioned four reasons that limit us in getting the real throughput of diagnostics implementation over Ethernet. It can be seen that one of the reasons is the result of the difference between CAN and Ethernet characteristics (bottleneck) and some are the result of

the current infrastructure and software architecture. In the current software infrastructure of the DoIP gateway, I can only send one request at a time on the CAN bus and wait for the response, I cannot send the next request before receiving the response of the last request, current software architecture that is implemented on DoIP gateway limit the throughput of the system. Designing the system in a way that let us have a multithreaded functionality can raise the performance and throughput of the system. I should mention that my work has just focused on the implementation of DoIP Server in a way that is consistent with current software architecture. Although I can receive multiple requests on a TCP frame of the Ethernet connection but I cannot send them on the CAN bus in the way that I mentioned earlier so that I can increase the throughput. This limitation is has been revealed after more investigation on the result.

6 Conclusions and Suggestions for Future Work

Implementation of the first step proves the possibility of ISO 13400 deployment on Scania's vehicles, while meeting current requirements [46] [47] [48]. There are some minor modifications required, which arise from the differences between CAN and Ethernet. These changes include length of address field, timing parameters, payload size and one specific change originated from manufacturer use case, i.e., the need to have a list of all ECUs stored in the diagnostic gateway.

The result of my practical investigation in chapter 5 shows that it is not practical to get the real bit rate that can be achieved by Ethernet due to lower bit rate of CAN. Moreover, the requirement of transferring a higher number of messages defined by DoIP increases the response time. On the other hand, not needing any VCI and plug and play compatibility are among some of the advantages that Ethernet brings for the system. Upon implementing the network scenario, it is possible to run remote diagnostics on a truck which is in a repair shop by providing an internet connection. This phrase is valid if, and only if, we introduce a new node to the system which is a DoIP gateway. Another option is making use of the current infrastructure with some modification for this system, since the investigation shows that there is one ECU which is capable of handling the job. This option is valid since in diagnostic applications we are not in need of quick response time. In fact the choice of whether to deploy DoIP or not, is a trade off, between bringing ease to the users i.e., those in the assembly line and repair shops or even the owner of the vehicle and longer response time.

Furthermore, by moving those ECUs that are in need of a higher bandwidth to the Ethernet network via the gateway and switch, the manufacturer can benefit from both real-time characteristics of CAN and the speed of Ethernet. In this case those ECUs that are on the Ethernet network are not going to suffer from the bottleneck imposed by CAN. The step by step approach and introduction of the model in two levels shows the complexity of the work and helps not to miss any possible bug in the system, while improving the system gradually. The proposed model demonstrates a way that provides the best solution for back ward compatibility by introducing modular and independent components.

6.1 Future Work

In this degree work we just focused on the investigation regarding risk and possibility and proposed models in two levels for implementation and tried to prove the proposed models by implementing them on the real hardware, which provided us with useful results. We also proposed a model for the second step and tried to prove our model theoretically and by referring to previous works. Simulation of this model and implementing it on real hardware is one of the essential steps before utilizing it on products. The concept of traffic shaping and congestion that are in need of any Ethernet network in order to guarantee real-time behaviour is one of the future tracks that can be taken for investigation of the model. There are already many industrial solutions in this regard like Ethernet Powerlink, EtherCAT just to name a few, which usually override the standards to meet some type of requirements of industry.

Based on the result on the investigation and being proved that DoIP is feasible on Scania's products, following tracks are suggested as a future work:

- ✓ Implementation of complete DoIP with all possible point to point and network scenarios on current hardware to see the overall performance that can be brought for

the system in real product. To do so, we need to handle the challenges that were discussed in section 4.1.

- ✓ Implementation of the current work on the green bus on a dedicated hardware as proposed in the first step model. Implementation of the model must be done for both first level and second level.
- ✓ We can have a big step and try to implement the second step as discussed in sections 4.3.2 and 4.3.4. The work can be done first as a simulation to find the probable obstacles and later implemented on a real hardware.

7 References

[1] G. Coulouris “Distributed Systems: Concepts and Design”, Addison Wesley, 2005.

[2] Tindell, K.W., Hansson, H., Wellings, A.J., “Analysing Real-Time Communications: Controller Area Network (CAN)”, Real-Time Systems Symposium, 1994.

[3] International Organization for Standardization,” Road Vehicle-Controller Area Network (CAN) – part 1: Data Link Layer and Physical Signalling”, ISO 11898-1, 2003.

[4] H.Chen, J. Tian, “Research on the Controller Area Network ”, International Conference on Networking and Digital Society, 2009.

[5] X. He, Q. Wang, Z. Zhang, “A Survey of Study of FlexRay Systems for Automotive Net”, International Conference on Electronic & Mechanical Engineering and Information Technology, 2011.

[6] N. Navet, Y. Song, F. Simont-Lion, C.Wilwert, “Trends in Automotive Communication Systems”, Proceedings of the IEEE, 2005.

[7] M. Lee, S. Chung, H. Jin, “Automotive Network Gateway to Control Electronic Units through MOST Network”, International Conference on Consumer Electronics (ICCE), 2010.

[8] M. Lee, H. Jin, “User-Level Network Protocol Stacks for Automotive Infotainment Systems”, International Conference on Embedded and Ubiquitous Computing (ECU), 2010.

[9] W. Jian-guo, Y. Bing, “Realization of Audio Transmission Node for Vehicular MOST Network”, International Conference on Control and Electronic Engineering (CMCE), 2010.

[10] H. Kopetz, W. Elmenreich, C. Mack, “A Comparison of LIN and TTP/A”, Proceedings 2000 IEEE International Workshop on Factory Communication Systems, 2000, pp. 99-107.

[11] U. Keskin, “In-Vehicle Communication Networks: A Literature Survey”, Technische Universiteit Eindhoven, 2009, http://alexandria.tue.nl/repository/books/652514.pdf.

[12] G. Cena, A. Valenzano, “Performance Analysis of Byteflight Networks”, Proceedings 2004 IEEE International Workshop on Factory Communication Systems, 2004, pp. 157- 166.

[13] J.R. Pimentel, T. Sacristan, “A Fault Management Protocol for TTP/C”, IEEE International Conference on Industrial Electronics Society, 2001, vol. 3, pp.1800-1805.

[14] J.Bell, “Network Protocols Used in the Automotive Industry”, Technical report, SoftFMEA, Aberystwyth University, 2002.

[15] SAE International Standard, “ J1850 Class B Data Communications Network Interface”, 2012, http://www. http://standards.sae.org/wip/j1850/.

[16] R. Bruckmeier, “Ethernet for Automotive Applications”, Freescale Technology Forum, 2010.

[17] L. Lo Bello, O. Mirabella, “Analysis and Comparison of Different Interconnection Solutions for Switched Ethernet Networks”, Proceedings 2000 IEEE International Workshop on Factory Communication Systems, 2000, pp. 221- 230.

[18] P. Pedreiras, L. Almeida, P. Gia, “The FTT-Ethernet Protocol: Merging Flexibility, Timeliness and Efficiency”,Proc. 14th Euromicro Conference on Real-Time Systems(ECRTS’02), 2002, pp. 134- 142.

[19] J.-L. Scharbarg, M. Boyer, C. Fraboul, “CAN-Ethernet Architecture for Real-Time Applications”, 10th IEEE Conference on Emerging Technologies and Factory Automation, 2005, pp. 244- 252.

[20] K. Tappayuthpijarn, B. Krebs, E. Steinback, R. Bogenberger, ”Traffic Shaping for Resource-Efficient In-Vehicle Communication”, IEEE Transations on Industrial Informatics, 2009, vol. 5, pp. 414- 428..

[21] T. Skeie, S. Johannessen, and O. Holmeide, “Timeliness of Real-Time IP Communication in Switched Industrial Ethernet Networks,” *IEEE Trans Ind. Inform.*, vol. 2, no. 1, pp. 25–39, Feb. 2006.

[22] J. Kiszka, B. Wagner, “RTnet- A Flexible Hard Real-Time Networking Framework”, 10th IEEE Conference on Emerging Technologies and Factory Automation, 2005, vol. 1, pp. 448- 456.

[23] H. Kopetz, A. Ademaj, P. Grillinger, K. Steinhammer, “The Time-Triggered Ethernet (TTE) Design” , 8th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing 2005, pp. 22- 33.

[24] H. Lim, L. Volker, D. Herrscher, “Challenges in a Future IP/Ethernet-Based in-Car Network for Real-Time Applications”, Conference on Design Automation, 2011.

[25] M. Rahmani, R. Steffen, K. Steinbach, G. Giordano, “Performance Analysis of Different Network Topologies for In-Vehicle Audio and Video Communication”, 4th International Conference on Telecommunication Networking Workshop on QoS in Multiservice IP Networks, 2008.

[26] M. Rahmani, K. Tappayuthpijarn, B. Krebs, E. Steinback, R. Bogenberger, ” Traffic Shaping for Resource-Efficient In-Vehicle Communication”, IEEE Transactions on Industrial Informatics, 2009, vol. 5, pp. 414- 428.

[27] S. Ruping, E. Vonnahme, J. Jasperneite, “Analysis of Switched Ethernet Networks with Different Topologies Used in Automation Systems”, CiteeSeer, 1999, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.40.8224.

[28] R.M. Daoud, H.H. Amer, H.M. Elsayed, Y. Sallez, “Ethernet-Based car Control Network”, Canadian Conference on Electrical and Computer Engineering, 2006.

[29] H. Lim, B. Krebs, L. Volker, P. Zahrer, “Performance Evaluation of the Inter-Domain Communication in a Switched Ethernet Based in-Car Network”, 36th IEEE Conference on Local computer Networks (LCN), 2011, pp. 101- 108.

[30] Charles E.Spurgeon, “Ethernet: The Definitive Guide” , 1st Edition February 2000.

[31] H.-E. Schurk, H.D. Fournell, “Onboard Diagnosis of Electronics, a Contribution to Vehicle Reliability”, 37th IEEE Conference on Vehicular Technology, 1987, pp. 343- 350.

[32] International Organization for Standardization, “Road Vehicle-Diagnostic System-Keyword Protocol 2000”, ISO 14230-1, Mar. 15, 1999.

[33] C.E. Lin, Y.-S. Shiao, C.-C. Li, S.-H. Yang, S.-H. Lin, C.-Y. Lin, “Real-Time Remote Onboard Diagnostics Using Embedded GPRS Surveillance”, IEEE Transactions on Vehicular Technology, 2007, vol. 56, pp. 1108- 1118.

[34] N.N. Hasan, A. Arif, U. Pervez, M. Hassam, S.S.U. Husnain, “Micro-Controller Based On-Board Diagnostic (OBD) System for Non-OBD Vehicles”, 13th International Conference on Computer Modeling and Simulation, 2011.

[35] H. Jie, Y. Fuwu, T. Jing, W. Pan, C. Kai, “Developing PC-Based Automobile Diagnostic System Based on OBD System”, Conference on Power and Energy Engineering, 2010.

[36] International Organization for Standardization, “ Road Vehicles-Unified Diagnostic Services (UDS) -- Part 1: Specification and Requirements”, ISO 14229-1, 2006.

[37] International Organization for Standardization, “ Road Vehicles -- Implementation of WWH-OBD Communication Requirements -- Part 1: General Information and Use Case Definition”, ISO 27145, 2011.

[38] E. Matzols, “Ethernet in Automotive Network”, Thesis report, Kungliga Tekniska Högskolan (KTH), 2011.

[39] SAE International Standard, “Surface Vehicle Recommended Practice”, SAE J1939, 2010.

[40] International Organization for Standardization, “Road Vehicles – Diagnostic Communication over Controller Area Network (DoCAN) - Part2: Transport Protocol and Network Layer Services”, ISO 15765-2, 2011.

[41] International Organization for Standardization, “Road Vehicles-Unified Diagnostic Services (UDS) -- Part 2: Session Layer Services”, ISO 14229-2, 2012.

[42] International Organization for Standardization, “Road Vehicles -- Diagnostic Communication over Internet Protocol (DoIP) -- Part 1: General Information and Use Case Definition”, ISO 13400-1, 2011.

[43] International Organization for Standardization, “Road Vehicles -- Diagnostic Communication over Internet Protocol (DoIP) -- Part 2: Network and Transport Layer Requirements and Services”, ISO 13400-2, 2011.

[44] J.-L. Scharbarg, M. Boyer, C. Fraboul, J. Ermont, “TTCAN over Mixed CAN/Switched Ethernet Architecture”, 10th IEEE Conference on Emerging Technologies and Factory Automation, 2005, vol. 2, pp. 664- 668.

[45] Scania Internal Document, CAN Overview SOP0905.

[46] Scania Internal Document, Technical Regulation, SCANIA ECU Requirements for UDS, TB4061.

[47] Scania Internal Document, Technical Regulation, Requirements on Diagnostic Services for KWP2000 in SCANIA ECU, TB4060.

[48] Scania Internal Document, Technical Regulation, Data Communication Requirements, Control Units Connected to a SAE J1939 Network Segment, TB4262.

[49] Scania Internal Document, C200 Programming Handbook.

[50] Fan, X. and M. Jonsson, "Guaranteed Real-Time Services over Standard Switched Ethernet" Proc. of the 30th Annual IEEE Conference on Local Computer Networks (LCN'2005), Sydney, Australia, Nov. 15-17, 2005, pp. 489- 492.

[51] Hoang, H., M. Jonsson, U. Hagström, and A. Kallerdahl, "Switched Real-Time Ethernet with Earliest Deadline First Scheduling - Protocols and Traffic Handling" Proc. Workshop on Parallel and Distributed Real-Time Systems (WPDRTS'02) in Conjunction with International Parallel and Distributed Processing Symposium (IPDPS'02), Fort Lauderdale, FL, USA, April 15-16, 2002, pp. 94- 99.

.

8 Table of Figures

9 Table of Tables

10 Abbreviations

C

CAN
Controller Area Network 5
COO
Coordinator 43
CSMA/CD
Carrier Sense Multiple Access/ Collision Detection 15
CSMA/CR
Carrier Sense Multiple Access/Collision Resolution 15

D

DoCAN
Dianostics over Controller Area Network 11
DoIP
Diagnostic over IP 5, 10
DTC
Diagnostic Trouble Code 25

E

ECU
Electronic Control Unit 5
EMC
Electromagnetic Compatibility 20

F

FTDMA
Flexible Time Division Multiple Access 17

I

IP
Internet Protocol 11
ISO
International Standard Organization 25

K

KWP2000
Keyword Protocol 2000 25

L

LIN
Local Interconnect Network 18

M

MAC
Medium Access Controller 15
MOST

11 Appendix

Following tables describe the response times related to DoCAN and DoIP regarding figure 26 and figure 27.

Table 9. DoCAN vs. DoIP

ECU_id	R_DoCAN	Serveice Id	N_b_CAN	N_b_Eth	UDS	KWP	R_DoIP	R_T_Eth_pure data(1000B)
ECU1	130	1902F1	787	799	1		267	2
ECU1	134	1902F1	787	799	1		253	
ECU1	129	1902F1	787	799	1		275	
ECU1	147	1902F1	787	799	1		302	
ECU1	147	1902F1	787	799	1		234	
ECU1	141	1902F1	787	799	1		270	
ECU1	149	1902F1	787	799	1		235	
ECU1	129	1902F1	787	799	1		256	
ECU1	128	1902F1	787	799	1		251	
ECU1	136	1902F1	787	799	1		286	
	139.57						262.9	
ECU1	137	1902F2	735	747	1		229	
ECU1	138	1902F2	735	747	1		232	
ECU1	132	1902F2	735	747	1		249	
ECU1	124	1902F2	735	747	1		247	
ECU1	123	1902F2	735	747	1		258	
ECU1	124	1902F2	735	747	1		263	
ECU1	122	1902F2	735	747	1		259	
ECU1	138	1902F2	735	747	1		258	
ECU1	157	1902F2	735	747	1		264	
ECU1	153	1902F2	735	747	1		247	
	134.8						250.6	
ECU1	16	190201	55	67	1		190	
ECU1	17	190201	55	67	1		190	
ECU1	16	190201	55	67	1		199	
ECU1	15	190201	55	67	1		211	
ECU1	16	190201	55	67	1		181	
ECU1	16	190201	55	67	1		187	
ECU1	17	190201	55	67	1		190	
ECU1	15	190201	55	67	1		207	
ECU1	18	190201	55	67	1		202	
ECU1	16	190201	55	67	1		191	
	16.2						194.8	
ECU2	38	1802FFFF	20	32		1	214	
ECU2	55	1802FFFF	20	32		1	203	
ECU2	34	1802FFFF	20	32		1	213	
ECU2	51	1802FFFF	20	32		1	210	
ECU2	49	1802FFFF	20	32		1	213	
ECU2	56	1802FFFF	20	32		1	223	
ECU2	34	1802FFFF	20	32		1	203	
ECU2	36	1802FFFF	20	32		1	213	
ECU2	47	1802FFFF	20	32		1	198	

ECU2	50	1802FFFF	20	32		1	211	
	45						210.1	
ECU2	9	1802000C	5	17		1	223	
ECU2	8	1802000C	5	17		1	245	
ECU2	7	1802000C	5	17		1	233	
ECU2	9	1802000C	5	17		1	231	
ECU2	9	1802000C	5	17		1	233	
ECU2	11	1802000C	5	17		1	233	
ECU2	7	1802000C	5	17		1	233	
ECU2	7	1802000C	5	17		1	244	
ECU2	8	1802000C	5	17		1	233	
ECU2	7	1802000C	5	17		1	238	
	8.2						234.6	
ECU3	48	1802FFFF	23	35		1	222	
ECU3	43	1802FFFF	23	35		1	221	
ECU3	44	1802FFFF	23	35		1	210	
ECU3	47	1802FFFF	23	35		1	209	
ECU3	43	1802FFFF	23	35		1	221	
ECU3	43	1802FFFF	23	35		1	216	
ECU3	45	1802FFFF	23	35		1	212	
ECU3	45	1802FFFF	23	35		1	221	
ECU3	50	1802FFFF	23	35		1	231	
ECU3	45	1802FFFF	23	35		1	221	
	45.3						218.4	
ECU3	44	1A81	42	54		1	211	
ECU3	47	1A81	42	54		1	220	
ECU3	48	1A81	42	54		1	219	
ECU3	41	1A81	42	54		1	200	
ECU3	49	1A81	42	54		1	221	
ECU3	50	1A81	42	54		1	222	
ECU3	44	1A81	42	54		1	222	
ECU3	46	1A81	42	54		1	220	
ECU3	57	1A81	42	54		1	208	
ECU3	44	1A81	42	54		1	222	
	47						216.5	

Table 10. Test Tool to DoIP Gateway

ECU_id	R_DoCAN	Serveice Id	N_b_CAN	N_b_Eth	UDS	KWP	R_DoIP	R_T_Eth_pure data(1000B)
Gateway	2174	1802FFFF	8	20		1	1714	2
Gateway	2230	1802FFFF	8	20		1	1630	
Gateway	2169	1802FFFF	8	20		1	1630	
Gateway	2203	1802FFFF	8	20		1	1664	
Gateway	2216	1802FFFF	8	20		1	1608	
Gateway	2151	1802FFFF	8	20		1	1605	
Gateway	2227	1802FFFF	8	20		1	1582	
Gateway	2128	1802FFFF	8	20		1	1519	
Gateway	2224	1802FFFF	8	20		1	1670	
Gateway	1878	1802FFFF	8	20		1	1658	
	2160						1628	
Gateway	1058	1803FFFF	44	56		1	1670	
Gateway	1068	1803FFFF	44	56		1	1665	
Gateway	1086	1803FFFF	44	56		1	1598	
Gateway	1072	1803FFFF	44	56		1	1578	
Gateway	1057	1803FFFF	44	56		1	1661	
Gateway	962	1803FFFF	44	56		1	1650	
Gateway	1053	1803FFFF	44	56		1	1642	
Gateway	1076	1803FFFF	44	56		1	1484	
Gateway	1075	1803FFFF	44	56		1	1586	
Gateway	1069	1803FFFF	44	56		1	1622	
	1057.6						1615.6	

12 Presentation of the author

Saeed Moradpour Chahaki

Graduated as bachelor student in Robotics engineering from Shahrood university in Iran, 2008. Graduated as master student in Embedded and Intelligent Systems from Halmstad university in Sweden, 2012.