Evaluation of OPC-UA Technology in a Car-2x Communication towards an Industry 4.0 Driven Automotive Domain

Master Thesis

Submitted in Fulfilment of the

Requirements for the Academic Degree

M.Sc. in Automotive Software Engineering

Dept. of Computer Science

Chair of Computer Engineering

Submitted by:	Nikith Swamy
DOB:	20.01.1989 (Shimoga, Karnataka, India)
Matr. Nr.:	367091
Supervising Tutors:	Prof. Dr. W. Hardt
	Jun.-Prof. Dr.-Ing. Alejandro Masrur
Supervisor (DLR):	Christian Linder
Date and Place of submission:	20.11.2017, Chemnitz

Acknowledgement

The successful completion of this Master Thesis work has been possible with the help and support of a lot of people. Therefore, I would like to take this opportunity to thank each and every person who has been involved.

I would like to begin by expressing my sincere thanks to my mentor Christian Linder at DLR, Braunschweig. He has been the one constant source of support and encouragement throughout the entire course of my thesis. Not only has he introduce me to the current state of the art at the research facility but has been patient enough to explain and help me get acquainted with the infrastructure. His innovative ideas in approaching the tasks have taught me a great deal. It was indeed an enriching and satisfying experience for me to be part of a project like this. He has been extremely understanding and has gone to great lengths to provide all the necessary help that I needed. I will always be grateful to him for giving me this opportunity.

A very special thanks to my supervisor, Jun.-Prof. Dr.-Ing. Alejandro Masrur at the Professorship of Computer Science, TU Chemnitz. Right from the start he has been patient and understanding in guiding me in the right direction to present my thesis work from the scientific research point of view. Through our various presentations and discussions, I have been able to learn a lot from him. He has taken a lot of effort to help me out in the documentation of my thesis work and help me correct all the shortcomings. Without his passionate participation and input, my Master Thesis work would not have been possible. I am extremely grateful to him.

Finally, I must express my very profound gratitude to my family and all my friends for always sticking by me and expressing their unfailing support and the never ending encouragement. This accomplishment would not have been possible without them. Thank you.

Sperrvermerk

Die nachfolgende Masterarbeit enth¨alt vertrauliche Daten der DLR. Ver¨offentlichungen und Vervielf¨altigungen der Masterarbeit - auch nur auszugsweise
- sind ohne ausdru¨ckliche Genehmigung der DLR nicht gestattet. Die Masterarbeit ist nur den Korrektoren sowie den Mitgliedern des Pru¨fungsausschusses zug¨anglich zu machen.

Abstract

In the area of Car 2X communication, ensuring a good quality secure communication always plays a vital role. One cannot ignore the dangers of losing data or getting data stolen from the public domain. This, especially when it involves the safety of the general public. Loss of data between two information exchange systems may lead to unwanted loss, additional overhead of fixing the devices among many others. In today's world there are a lot of car companies that define information exchange models based on their language of programing and also based on the underlying architecture the products of the company are built in. therefore there is always a need to ensure a good communication while ensuring security.

Current state-of-the-art techniques involved in the transformation of signals and information between a car and an infrastructure or a machine to machine have the underlying problem of security. The application platform for Intelligent Mobility can be considered as the most advanced form of communication architectures used on the roads today. However, there will always come another better technology as time goes by.

The thesis "Evaluation of OPC-UA Technology in Car2X Communication towards an Industry 4.0 Driven Automotive Domain" has been done at DLR, Braunschweig along with the Faculty of Science in Chemnitz University of Technology. The purpose of this thesis is to bring about a technology into the automotive domain, which can be made into a standard for security and robustness. The layered chapters of this thesis researches into the possibilities of an alternate to the current standards. The application is implemented by the use of a server-client model for information exchange. The evaluation is done based on the results and the huge potential OPC UA carries in the automation industry.

KEYWORDS: (Car2X, Industry 4.0, AIM, Vehicle to Infrastructure, Mobile Road Traffic Signal, OPC-UA)

Nomenclature

AIM	Application Platform Intelligent Mobilitat
ASN.1	Abstract Syntax Notation One
AU	Application Unit
BER	Basic Encoding Rules
BSA	Basic Set of Applications
BTP	Basic Transport Protocol
C2C	Car-to-Car
C2C-CC	CAR 2 CAR Communication Consortium
C2I	Car-to-Infrastructure
C2X	Car-to-X
CA	Cooperative Awareness
CAM	Cooperative Awareness Message
CCH	Control Channel
CCU	Communication Control Unit
CEN	Comit'e Europ'een de Normalization
COM	Component Object Model
CSM	Cooperative Speed Management
CSMA/CA	Carrier Sense Multiple Access / Collision Avoidance
DENM	Decentralized Environmental Notification Message
DLR	German Aerospace Centre
ETSI	European Telecommunications Standards Institute
GPS	Global Positioning System
HMI	Human Machine Interface
I2V	Infrastructure-to-Vehicle
IEEE	Institute of Electrical and Electronics Engineers
INS	Inertial navigation system
IP	Internet Protocol
ITS	Intelligent Transport System
ITS S	ITS station
JAR	Java Archive
JDK	Java Development Kit
LDS	Location Discovery Server

LSA	Traffic signal
MAC	Media Access Control
RSU	Roadside Unit
SDK	Software Development Kit
SCH	Service Channel
SHB	Single-hop broadcast
TOPO	Topology Specification
UDP	User Datagram Protocol
UTC	Coordinated Universal Time
V2I	Vehicle-to-Infrastructure
V2X	Vehicle-to-Vehicle
V2V	Vehicle-to-X
VANET	Vehicular Ad-hoc Network
WAN	Wireless Local Area Network
XML	Extensible Markup Language

Contents

List of Figures

List of Tables

1 Introduction

1.1 Industry 4.0

Germany has one of the most competitive production and manufacturing industries in the world today. This success is attributed to the substantial efforts in investments in the research and development areas, as well its ability to manage complex processes where tasks are executed by various partners, across many locations. Recent advances in the manufacturing industry have paved way for a systematic deployment of connected systems, within which information from corresponding perspectives is closely monitored and synchronized between the physical factory floor and the cyber computational space. Moreover, with the help of advanced information analytics, it is possible that networked machines will be able to perform more efficiently, collaboratively and resiliently [1].

Towards the end of the 18th century, mechanical manufacturing equipments were introduced and thus began industrialization. The first mechanical loom was designed in 1784. This revolutionized the way goods were manufactured. Steam engines were introduced which drove the looms. The result was an extreme increase in productivity thus adding a new dimension to the industrial output. This marked the first industrial revolution. The second industrial revolution followed a hundred years later, around the turn of the 20th century. This era introduced electrically-powered mass production based on division of labor. This was then succeeded by the third industrial revolution that began in the late 1970s.

The third industrial revolution was all about automation and digitization of the technology. Automation streamlined and smoothened the entire industry. This increased the overall efficiency and brought along with it, a new dimension in industrial output. Digitization helped fuel the efficiency even more as everything could now, be controlled by the touch of a button or readings could be read on screens and remotely. We are now in a very exciting time where the fourth industrial revolution has kicked off, “Industry 4.0”. This term was introduced by the German Federal Ministry of Education and Research which involves the technical integration of Cyber-Physical Systems (CPS) into logistics and manufacturing and the usage of Internet of Things and Services (IoTS) in industrial processes. The different revolutions are as shown in Figure 1.1 [2].

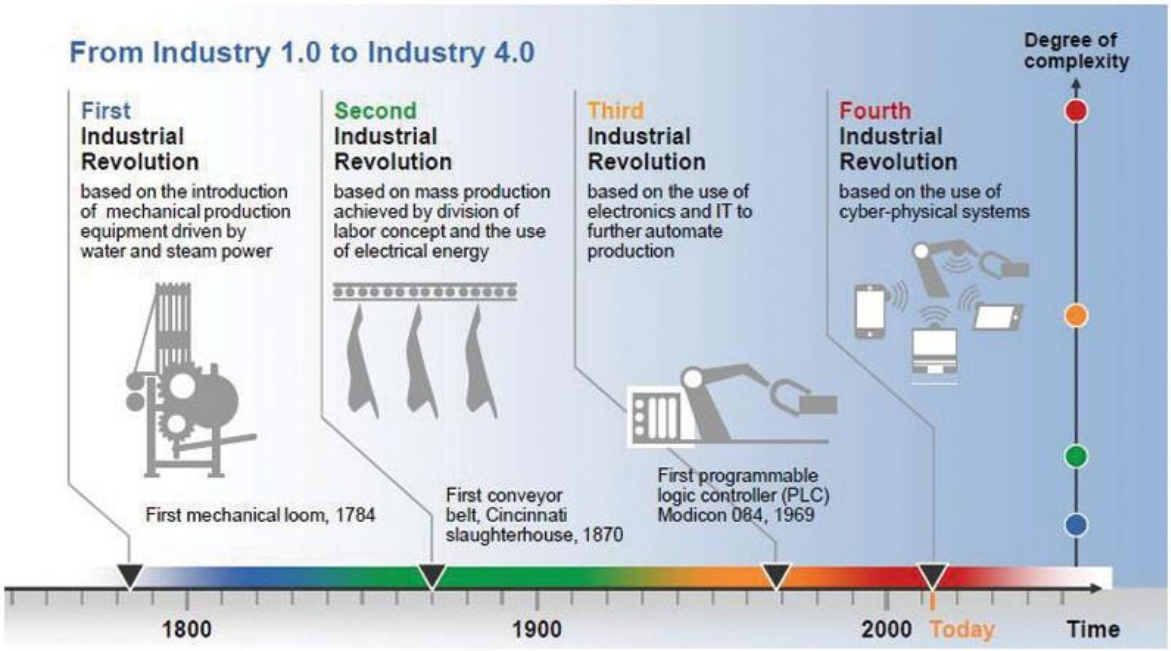

Figure 1.1 : Transition from the First Industrial Revolution to Industry 4.0 [2]

In the industry 4.0, sensors, machines, work pieces, and IT systems are connected along the value chain beyond a single enterprise. These connected systems (also referred to as CPS) interact with one another with the help of standard Internet-based protocols and are used for data analysis for failure detection while having the ability to configure themselves and adapt to changes. By doing so, Industry 4.0 makes it possible to gather and analyze data in a much faster way ensuring more flexibility and efficiency. This in turn helps produce higher quality goods at reduced costs. Moreover, new business models and production processes are adopted which enable a new level of mass customization as more industrial producers invest in Industry 4.0 technologies to enhance the quality of their services [3]. This results in a standard which can be acceptable and practically usable by all businesses working in similar fields.

Some of the technological trends that form the foundation for Industry 4.0 are as follows [3]:

1. **Big Data and Analytics** - A relative and economic concept which has been around for quite some time, Big Data means exhaustive collection of data sets from many different sources of varying sizes. In today's world, some organizations deal with massive data volumes while others have smaller data sets but have a broader sets of formats and sources. They are primarily used for quality optimization of

products, energy-saving and failure detection. High-performance analytics are used to analyze these massive amounts of data thereby determining which data is useful and critical and which is not.

2. **The Industrial Internet of Things** – The main idea behind the Internet of things, is to bring data from many sources and intelligent machines together, from people around the world. This helps the manufacturers and producers boost their productivity and efficiency. It also helps in generating revenue from new sources. With The Industrial Internet of Things, more devices are connected together and can communicate with each other with real-time responses. The combination of the technical standards of Industry 4.0 and Industrial Internet Consortium (IIC) gave rise to the term, ”Connected Industry” which paved the way for numerous new cross-border business opportunities for Industry 4.0 solutions [4]. For example, companies relying on predictive maintenance helps them in avoiding unnecessary shutdowns and keep the products and services flowing.

3. **The Cloud -** With Industry 4.0, the amount of data shared across many plat- forms by companies using cloud-based software can improve the functionality of the technology within a matter of milliseconds. Analytics applied to the cloud storage can help determine inconsistencies which need immediate attention. Data is stored in the cloud to improve its accuracy and reliability. This directly leads to more flexibility when reacting under foreseen and unforeseen circumstances. Also, systems for monitoring and controlling processes can be made cloud-based. Companies selling Manufacturing Execution Systems (MES) can be deployed on the cloud as well.

4. **Cybersecurity** - Based on the current ”Connected Industry” paradigm, cyber-security is a very important prerequisite needed for manufacturers and production equipment suppliers. This is in particular to prevent any form of attack from cyber threats or unauthorized access to production systems. The need to protect environments, critical industrial systems and manufacturing systems increases dramatically with the use of standard communication protocols and improved connectivity in Industry 4.0. As a result of this, encrypted and reliable communication are very essential.

1.1.1 Internet of Things (IoT)

The Internet of Things (IoT) is a recent communication paradigm that envisions a near future, here the objects of everyday life will be equipped with microcontrollers, transceivers for digital communication along with suitable protocol stacks that will make them able to communicate with one another. This will become an integral part of the internet [7]. This concept aims at making the internet furthermore immersive and pervasive.

NO/CLOSED NETWORKS	INTERNET OF THINGS			
	Endpoints	Simple Hubs	Integrating Hubs	Enhanced Services
Optimize			GE Software Predix and other industrial platforms for interconnecting analytics engines and business operations Large-scale digital city systems like those under development at MIT and in Barcelona	
Adapt	Stand-alone GPS navigation devices	Progressive Snapshot and other auto insurance telematics systems Smartphone apps that use location tracking	Apple HomeKit and other protocol-based platforms allowing diverse devices in a building to interconnect to one another and the Internet	Emerging systems for setting insurance rates based on health and driving behavior
Control	Motion- or light-responsive alarms and controls	Google Nest and other Internet-connected systems for heating, cooling, and ventilation Estimote Beacon, iBeacon, and other Bluetooth-enabled object identification sensor systems	WeMo and other systems for controlling lights and appliances through remote or mobile devices	Potential connected-car traffic management systems
Monitor	Simple thermo-stats and motion sensors	Jawbone UP, Fitbit, and other fitness activity sensors and hub systems	BodyGuardian and other medical wearables that feed data to online diagnostic platforms	

Figure 1.2: Services available through the Internet of Things [5]

When it comes to smart manufacturing, the IoT becomes a huge aspect. The inter-connected network of IoT, Big Data and Cloud results in more sharing of data from sensors and connect to physical objects and people alike. These objects are connected to the sensors which are also connected to each other and simultaneously data will be gathered. The data is sent to shop floor workers, plant managers and different levels of the supply chain.

To deliver different products and services, a combination of five major technologies are required as the complexity of the devices and their connectivity increases. They are listed as follows [5]:

1. **Endpoints** - They are single-purpose sensors and actuators which record any changes in surroundings and provide any form of feedback corresponding to the change in data. They perform two critical tasks - gather and analyze data and reach out through the Internet to control objects.

2. **Simple Hubs** - They are devices which connect different endpoints to broader networks. They act as a joining point for a sensors and actuators which are located near each other.

3. **Integrating Hubs** - They are relatively complex devices that connect simple hubs and provide a diverse array of services that are fitted together in a seamless manner.

4. **Network and Cloud Services** - These provide the infrastructure required for IoT which can either accessible to the public or protected behind the firewall of a organization. Seamless connections are provided by these services to the Internet that simple and integrating hubs require along with the cloud computing services that are necessary to gather and analyze massive amounts of data from different endpoints. They also provide the infrastructure required to build or connect to social networks so users of IoT share data among them.

5. **Enhanced Services** - They are a growing category who deliver interactive functions with the help of information that is gathered and analyzed.

These technologies provide a vast array of diverse features for organizations who build IoT businesses.

1.2 Car-2X Communication

Car-to-X (C2X) is nothing but a means of communication that includes exchange of data and information between a vehicles and the transport infrastructure (C2I/V2I/I2V) or between vehicles (C2C/V2V). The basic goal is to give the driver an early and effective notice about any critical or dangerous situations along the road. Furthermore, increasing the traffic efficiency through a method of co-operative assistance. This type of communication is thus considered as the future of road transport throughout the world. In Europe, for example, there are different projects that essentially look to increase the safety in traffic as well as finding solutions to optimizing the road traffic. This is achieved by using C2X communication within the context of seeking applications with Intelligent Transport Systems (ITS). The information exchange takes place by a communication

within the vehicle ad hoc networks (VANETs) on Dedicated Short Range Communication (DSRC). In Europe, the term ITS-G5 is used to avoid the confusion between the US and the Japanese version of DSRC [ETS 12]. The communicating nodes represent the ITS-S stations (ITS-S), where vehicles are referred to as Vehicle ITS-S and the infrastructure as Roadside ITS-S2 [ETS-10a].

The infrastructure represents static road facilities and facilities on traffic routes that actively process and disseminate information. This is done by sending messages that can be received and expanded by the car. The car on the other hand processes the application data from the Adaptive Driver Assistance Systems (ADAS) and communicates with nodes in their environment. Car-2X communication thus represents the highly dynamic topology of the network which consists of short connection times, changing environmental conditions and directions of movement of vehicles. This forms the basis for the underlying communication system.

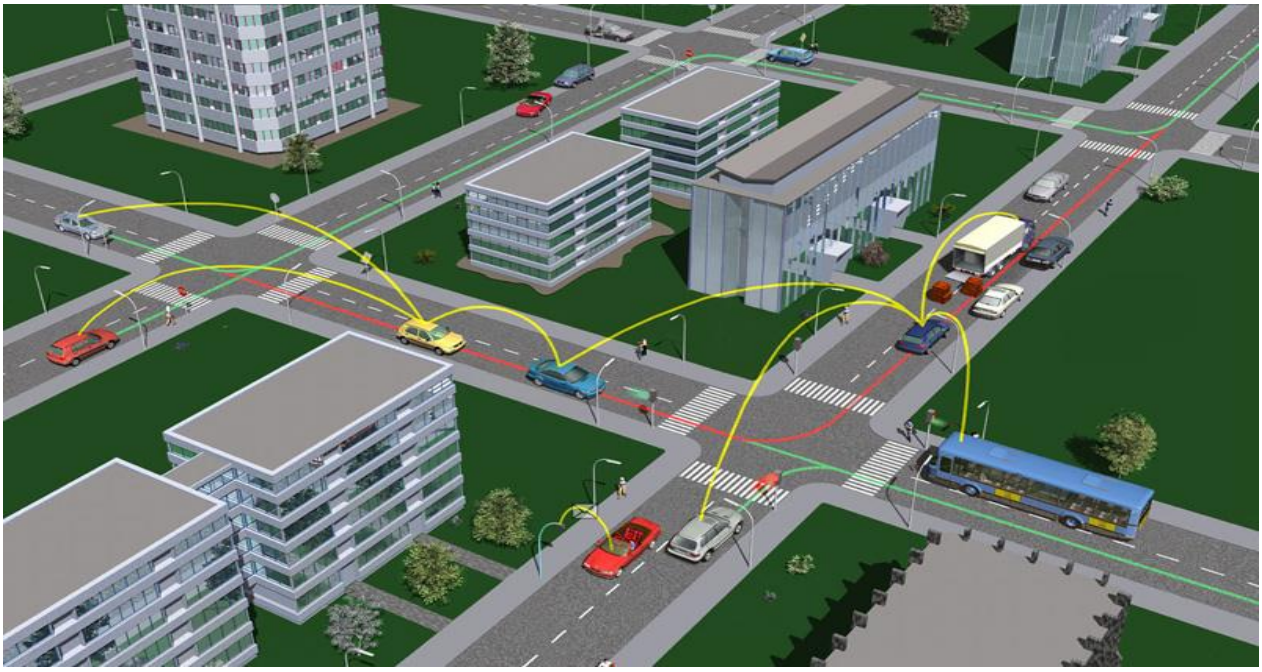

Figure 1.3: Car to Car city scenario [8]

The C2X technology also offers new possibilities to enhance road traffic safety and traffic efficiency at a large scale. In figure 1.3, the car is seen to be able to connect with cars beyond its horizon. This extended horizon means the potential limit on the area of coverage for the communication can be effectively extended as a network. It can no longer be limited to the horizon of the camera active in the car. This involves the usage of Radar and Lidar technologies. With the help of these technologies, when a dangerous

situation is detected along the road, the cars can send signals via hops. This allows the drivers behind to react on time and adapt their driving behavior accordingly.

C2X communication is based on the IEEE 802.11p standard which will be detailed in the following chapter. It is a standard that allows time critical safety applications at a very low data transmission delay. The European commission, in this context has agreed and initiated an allocation of the 30 MHz spectrum, from 5.875 to 5.905 GHz respectively. According to the European profile for safety related communications, two channels with 20 MHz each are assigned within the 802.11p standard. One channel is dedicated to network control and safety purposes and the second for safety applications [9].

1.3 Objective of the Thesis

The primary objective of the thesis is to evaluate the use of an automation industry communication technology called OPC-UA. This technology holds huge potential in fulfilling the ever evolving needs of a network used for communication in the public welfare sector. The second objective of this thesis is to implement a server client architecture within the existing state of the art communication model for Car2X communication and conclude the overall feeling of effectiveness of the new technology.

The Mobile Road Side Traffic Signal used today has an architecture for sending and receiving signals on a real time basis. However, it must be noted that the messages have weak security layer wrapped around them. The messages are not encrypted and may pose to be an easy target for potential hackers.

My thesis at DLR attempts to addresses this issue. The topic pertains to a vast field which needs to be researched deeply. The possibility to scratch the surface and trying to solve the issue within communication models, is always an interesting yet a daunting task. The usage of a server client architecture to send and receive messages has been implemented to share confidential information. Both the server and the client are interoperable which means both can either send or receive data as per the needs of the application. This communication technique is widely used in the automation industry but surprising not in the automotive industry. OPC UA thus might have huge potential in the Car2X communication models.

OPC also, finds its usage in machine to machine communication since it is basically an interpreter between machines. In Layman’s terms it can be considered as a language translator. This helps machines talk to each other. However, the biggest advantage of this system is within the underlying security feature. The OPC UA provides amongst others, its own layer of security which can be used to our advantage in the

communication between the mobile road side unit and the cars or any other machine. In addition to this, it leaves an option of adding an own layer of security which is of the developer’s choice. This thesis makes a humble attempt to foray into a new communication sector while having the ability to remain as a standard for communication for Car2X or machine to machine communication.

2 Literature Review

The history of manufacturing has always relied on automation and streamlining of process and tasks related to it. This has been extended into the software domain as well. The various software used in the current world heavily rely on the technologies that can run by themselves with the least human interaction. In other words, automation. The self-learning and communicating options are explored in this thesis with a focus on one such technology called OPC-UA. In this chapter we will discuss about two main parts. The first being a much more depth into the Car-2X communication with the description on the consortium. It also sheds light on the various standards and protocols that are being used in the industry within the topic of Car-2X communication protocols. The latter part of the review describes in detail, the motivation for OPC-UA, the architecture, specifications and the various layers relevant within the scope of this thesis.

2.1 Standardization and Committees

The following section introduces the main parties involved in the use and development of standardization of C2X communication systems in Europe. The unambiguous addition of responsibilities in addition to requirements and features allows a uniform standardization of required structures.

2.1.1 Car 2 Car Communication Consortium

The Car 2 Car Communication Consortium (C2C-CC) is a non-profit organization initiated by the European Vehicle manufacturers. This is achieved in cooperation with other car manufacturers, suppliers, research institutes and other partners. Their main aim is to increase the usage of C2X communication for road traffic safety.

The consortium is the authority throughout Europe, for the use of the frequency range between 5.875 to 5.905 MHz and is working towards creating a standard research practices as well as establishing the European standard for C2X communication systems in the future. The standardization is carried out in close cooperation with the European Telecommunication Standards Institute (ETSI) and the Comité Européen de Normalisation (CEN). In addition to this, there is also a coordination with other economic regions such as the USA and Japan. This allows a global harmony where hardware which is usable throughout the world is created. The other functions of the development of the

European ITS standards is to form a pre-requisite for active, co-operative applications to further increase the traffic safety and traffic efficiency. It is also extended to the infotainment services. The Car to Car Communication Consortium also defines a deployment strategy which helps in the market launch, for example the market launch in 2015 for the necessary applications and services. For the first phase in this strategy, there were ten one-day applications aimed at the introduction of a base system which had a limited complexity specified. These are used in the safety and non-safety domains respectively. The main components of the C2C-CC for Car-2X communication has been described in [CAR07]. The basic requirements for different applications are all explained in addition to the system requirements and restrictions.

Advanced developments are already being validated by the way of field trials in projects such as simTD[1] or DRIVE C2X[2]. The conceptual explanation provided also sheds light on the benefits of cooperative ITS.

2.1.2 ETSI

The European Telecommunications Standards Institute (ETSI) is a non-profit standardization organization. It is officially recognized by the European communities as best European Organization for Standardization and aims to create globally applicable standards for information and communication technologies. In the area of co-operative ITS, ETSI among others, is responsible for standardization of concepts of C2C-CC as listed in 2.1.1. On one hand, the communication architecture and the basic components are involved as in [ETS10a], on the other hand is the Basic Set of Applications (BSA) which contains a basic set of application dates that are within a three year period of completion can be used for standardization [ETS09].

ETSI is supported in the implementation of a variety of companies who actively participate in the standardization work. These include automobile manufacturers such as Volkswagen, Daimler and Renault, together with representatives of the automotive supply industry such as NEC, Continental or Hitachi. Likewise, chip manufacturers such as Freescale, Cohda Wireless and Autotalks, various network operators and research facilities are involved.

[1] http://www.simtd.de/

[2] http://drive-c2x.eu/

To speed up standardization and reduce the time to market, ETSI regularly organizes so-called plug tests, which serve to check the interoperability of the systems. On the one hand, these provide important feedback for the ETSI standardization bodies, and on the other hand, they provide solid and appropriate standards for the companies whose products are being tested at the event.

2.1.3 CEN

The Comité Européen de Normalisation (CEN) is one of the European Union's officially recognized standardization organization, which for European standards pertains to all technical areas other than electrical engineering and telecommunications. The CEN is a merger of 33 national standardization organizations which in turn provides a platform for the development of European standards and other technical documents relating to different types of products, materials, services and processes.

The Technical Committee CEN / TC278[3] is responsible for the development of European standards and technical specifications in the field of Intelligent Transport Systems (ITS), with the aim of ensuring interoperability between the different countries and ensuring technical interoperability to harmonize solutions. Working group WG16 deals with the standardization area of cooperative ITS.

2.2 Standards and Protocols

The following segment entails the standards and protocols used in the thesis. These standards also forms a basis for uniform exchange of information between communicating partners. These exchanges otherwise would have been very complex and difficult to accommodate.

2.2.1 IEEE 802.11p Standard

The IEEE 802.11p standard is an addition to the IEEE 802.11 standard to support the already established wireless technology in motor vehicles and is a reliable interface known for the usage in ITS applications. IEEE 802.11p introduces the key technology for DSRC, in a place like Europe especially for ITS-G5 within the 5.9 GHz range [ETS12]. The 802.11p extension is now part of the IEEE Standards 802.11-2012 [IEE12]. For better

differentiation to other extensions however, the 802.11 WLAN standard continues to be used as 802.11p in this work.

The standard data exchange is defined over ad hoc networks between vehicles, as well as between vehicles and the infrastructure. These communication links usually only exist for a short period of time. Therefore, in 802.11p, it is possible to exchange data without compelling a Basic Service Set (BSS) build [IEEE 12]. Consequently, it is not necessary to await for the completion of the connection establishment and the authentication process before exchanging the data. These security mechanisms need to be implemented in the higher levels of protocol.

IEEE 802.11p defines the physical layer and parts of the data link layer, especially the Media Access Control Layer (MAC), and the OSI reference model (see Annex A). The physical layer is similar to the 802.11a extension of the WLAN standard. However, in 802.11p, channels with 10MHz bandwidth are used in the 5.9GHz frequency band (see Table 1). Hence, half the bandwidth or twice the transmission time is used for a data symbol compared to 802.11a. This allows a more robust transmission among the specoal features of the radio channel in vehicle communication environments. For example, in the reflections of other cars or homes.

The ETSI is responsible for laying and reserving the ITS-G5A frequency band for ITS traffic safety applications and the ITS-G5B frequency band for the non-safety ITS traffic applications. Both areas are exclusively based and used with ITS-G5 compliant stations. In addition, the ITS-G5C band is used for broadband radio access networks (BRAN), Radio Local Area Network (RLAN) and WLAN. ITS-G5D is intended for the future use of ITS traffic applications [ETS13].

The IEEE 802.11p standard supports multi-channel operation for the co-existence of safety and non-safety applications. ETSI distinguishes the Service Channel (SCH) and Control Channel (CCH) [ETS13]. The typical usage of G5-CCH and G5-SCH1 to G5-SCH2 are basically reserved for ITS traffic safety. G5-SCH3 to G5-SCH5 are essentially incorporated for ITS traffic [ETS12].

As a modulation method, the 802.11p extension uses Orthogonal Frequency Division Multiplexing (OFDM) at the data rates of 3 to 27 Mbit/s [IEEE12]. Table 1 shows the Channel Assignment for the European region.

The 802.11p MAC layer is the media access control mechanism for Enhanced Distributed Channel Access (EDCA) which is ported from the IEEE 802.11e extension. This is based on the Distributed Coordination Function (DCF) and adds on to them with QoS attributes [IEE12]. DCF uses a principle for collision avoidance which is Carrier Sense Multiple Access / Collision Avoidance (CSMA / CA). The EDCA also defines

several traffic classes for the data traffic management (Traffic Classes). It uses a priority based system where high priority traffic receives a higher chance of being sent than a low traffic priority. Here, high priority traffic gets a chance to be sent, as traffic priority prior to sending a packet on average waits a little shorter than a station with lower priority traffic. This is done by short waits with the Arbitration InterFrame Spaces (AIFS) realized for higher priority packages. Furthermore, the QoS extension defines eight User Priorities (UPs), which are assigned to four different Access Categories (ACs) or queues [IEE12]. For ACs, the default values are specified for the minimum or maximum size 6 of the contention Windows (CW) and the number of AIFS. In summary, this leads to the drivtion of values shown in Table 2.

	Channel Type	Frequency Range [MHz]	IEEE Channel Number
ITS-G5A	G5-CCH	5895 to 5905	180
	G5-SCH2	5885 to 5895	178
	G5-SCH1	5875 to 5885	176
ITS-G5B	G5-SCH3	5865 to 5875	174
	G5-SCH4	5855 to 5865	172
ITS-G5C	G5-SCH7	5470 to 5725	94 to 145
ITS-G5D	G5-SCH5	5905 to 5915	182
ITS-G5D	G5-SCH6	5915 to 5925	184

Table 1: Survey of European Channel Assignment [ETS13]

However, the 802.11p extension differs from 802.11e in several important aspects, including the lack of the Hybrid Coordination Function (HCF) and the Transmit Opportunity (TXOP) functionality. IEEE 802.11p is special because it allows multichannel operation and sharp disaggregation between traffic classes.

AC	CW_{min}	CW_{max}	AIFS [µs]
AC VO	3	7	58
AC VI	7	15	71
AC BE	15	1023	110
AC BK	15	1023	149

Table 2: Overview of AIFS and CW sizes of the 802.11p Access Categories [ETS13]

2.2.2 UDP

The User Datagram Protocol (UDP) is a connectionless protocol that is within the Transport layer of the OSI reference mode (see Annex A). UDP requires minimal overhead but cannot provide end-to-end control for the transfer of data. Thus, it cannot be guaranteed that sent packets arrive in the same order in which they were sent or if the packet which is sent arrives at the receiver the first time. UDP is used to exchange datagrams between application processes. Ports are used to mention the address of the destination. Applications that use UDP need to be resilient towards any lost or unsorted packets, or provide of their own means for corrective actions and backups when needed. Data protection is not possible with UDP communication. UDP offers the possibility of an integrity check on the basis of a checksum, but a faulty transmission can hence only be recognized but not restored.

Unlike the Transmission Control Protocol, the data exchange is faster in UDP communication since there is no connection established between the sender and receiver prior to the data transfer. This becomes relevant especially in applications where small amounts of data are exchanged. Furthermore, the unsecured transmission offers an advantage in less fluctuations in transmission delay. With TCP, lost packets are automatically re-requested, this will cause a fluctuation in the total transmission time. In the case of connectionless protocols, however, lost packets do not delay the entire transmission, but only diminish the quality. As a result, UDP is especially suitable for applications which rely on a fast, efficient transmission. For example, real-time and multimedia applications and also servers that answer small requests from a large number of clients.

2.2.3 Car-2-Car Protocol stack

The Car2Car protocol stack is developed by C2C-CC and ETSI standardization committees [CAR07, ETS10a].The protocol of the architecture defined here follows the principles of the OSI reference model (see figure 2.1). The lowest level represents the access layer in the OSI model. Lying over this layer are the network and transport layers in accordance with the OSI model. Layers five to seven are formed by the facilities layer and the application layer [ETS10a]. Basically, the Car2Car protocol stack is a dual stack in which a subdivision into traffic safety, traffic efficiency or infotainment applications takes place. The access layers for traffic safety applications are based on the IEEE

802.11p standard. Infotainment applications access the traditional protocols and can use the ad hoc and multi-hop capabilities of the underlying Geo Networking protocol as and when needed. However, in traffic efficiency applications, both types of access can be provided depending on the requirement used in the scenario [CAR07].

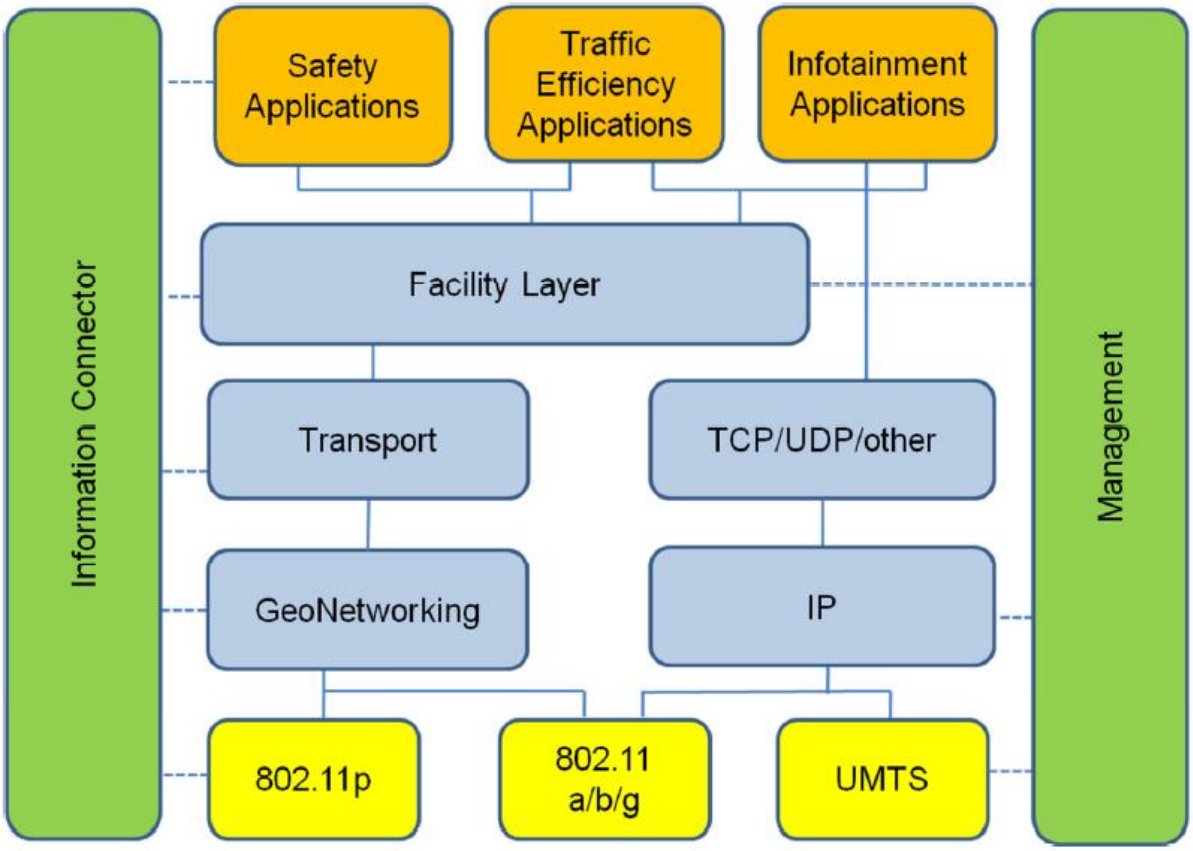

Figure 2.1: Protocol Architecture [NEC 11]

GeoNetworking is a network layer protocol that provides wireless multi-hop communication in an ad hoc network. The addressing and routing are made based on the geography [CAR07]. The main functions are the beaconing, location service and the forwarding of data packets [ETS11a]. The beaconing is primarily used to inform neighboring stations of their current location. Forwarding data packets involves the processing of messages in a node and also sending it to other nodes. GeoNetworking is based on the fact that each node knows its own geographic position, for example the GPS receiver uses a location table for the geographical location and also has information

regarding the other nodes. It also supports point-to-point and point-to-multipoint communication. The point-to-multipoint can be considered as group communication in which the end points are located within a geographic region.

The GeoNetworking protocol is different and has four basic forwarding types in Geographical Unicast (GeoUnicast), Topologically Scoped Broadcast (TSB), Geographically Scoped Broadcast (GBC) and Geographically-Scoped Anycast (GAC) [CAR07]. With GeoUnicast, a unidirectional transmission takes place from a transmitter to a single receiver, if and when necessary the information can be pushes further using hops. TSB is used to transport data from a sender to all nodes in the scope within the ad hoc network. The relevant area is limited by the hop limit, which means the number of possible hops has an upper cap. A special usage of the TSB is the single-hop broadcast (SHB), in which the hop limit amounts to one and the message is thus only transmitted to direct neighbors in the network. GBC is used to transfer data from one node to all nodes in a geographic target area. This area is determined by a geometric shape, for example in the shapes of circle and rectangle. The transmitter itself does not have to be in the target area because the message can be forwarded via other participants (hops). Unlike the GBC, the GAC does not have a data packet within the geographical area when the packet reaches the target. Hence, the data packet will only be transmitted to the receiver in the target area. TSB and SHB are commonly referred to in the context of broadcasting while GBC or GAC is referred with the term geocast.

According to the architecture of the Car2Car protocol stack, the GeoNetworking protocol provides the services of the access layer and in turn makes the parent transport layer services available. The Basic Transport Protocol (BTP) which is used, is comparable to UDP. The BTP provides connectionless transport in the ITS ad hoc network. The main task is the multiplexing of messages of different processes within the Facilities Layer which enables the transmission of packets via the GeoNetworking protocol. It also performs the demultiplexing at the destination [ETS11b]. The multiplexing or demultiplexing of messages is based on ports which represent the communication end points of the respective ITS protocol. This holds good for both at the source and the destination. Unlike UDP, BTP does not support either a long field or a checksum check [FBZL09].

The Facilities Layer provides the functionality of the Car2Car protocol stack which is available in the top three layers of the OSI [ETS10a]. It offers support for ITS applications that have generic functions. These functions also have to comply with the functional and operational requirements. The [ETS10a] also describes various other facilities. This includes, among other things, the data representation, i.e. the coding and

decoding of messages according to a formal language. For example, with ASN.1 and with the selection of the addressing mode on deeper layers. Another key feature is the time and position support, which provides information about the current time and geographic location of the ITS station. This includes the longitude, latitude and the altitude. A relevance check makes it possible to assess whether a received message is relevant to the context in question or not. In addition, the Facilities Layer implements a common message management that allows data exchange between ITS applications, such as Cooperative Awareness Messages (CAM) or Decentralized Environmental Notification Messages (DENM). A suitable communication interface for the message transmission is selected, by means of the channel selection support. The application layer of the Car2Car protocol stack is ready for all the common tasks of the Application services. This includes, among other things, the processing of messages and local vehicle sensor data as well as the interaction of the applications with people present within the vehicle or the Vehicle sensor data [CAR07].

2.2.4 ASN.1

The Abstract Syntax Notation One (ASN.1) is a description language for the definition of data structures which defines their implementations in one uniform format. It represents an abstract syntax notation at the layer level of the OSI reference model and is used to describe data types without going into the computer-internal representation. Using ASN.1 and a common coding rule, systems with different internal data representations can exchange messages among themselves.

As a rule, data specified in ASN.1 are Basic Encoding Rules (BER). In areas where space-saving coding is desired, for example, in the case of radio transmissions, the Packed Encoding Rules (PER) are preferred. ASN.1 encodings are smaller than many competing notations and thus allow a fast and reliable transmission of data, which in particular offers an advantage for wireless communication. An ASN.1 definition can be easily mapped into a C, C ++, or Java data structure which can then be used by the application code. With the help of runtime libraries it is possible to code and decode representations, for instance the XML format.

2.3 OPC-UA

In the early nineties, the use of personal computers with software for automation began rapidly in the industry. These computers at the time ran mostly windows operating systems and they helped in mainly visualization and control purposes. In the past years, one of the major efforts for the development of standardized automation software was the access to automation data. These were particularly with devices where a complex system of different bus systems, protocols, and interfaces were used. A lot of devices which connected to each other depended on the device drivers to function rather than depending on the application developers. As a result of this, there was a lot of additional overhead in getting all the devices to work together [19].

Since vendors of Human Machine Interface (HMI) and Supervisory Control and Data Acquisition (SCADA) faced similar issues, a task for was created in 1995 to standardize the same. The goal of the task force was to define a Plug and Play standard for device drivers providing a standardized access to automation data on Windows-based systems. As a result, in 1996 the OPC Data Access Specification was released as a solution. The term OPC was born from Object Linking and Embedding (OLE) for Process Control (OPC) [10]. OLE is a technology that lets you share data between two applications or between multiple applications. It was introduced by Microsoft. Here an object is used as a pointer to a data in a source file. Each time the source file is moved or deleted, the linked object that points to this file has to be re-created. A copy of this source file is an embedded object. In other words, linking establishes a connection between two objects, and embedding facilitates application data insertion [11].

The OPC Foundation is a non-profit organization that maintains this standard. Soon after this, all the members who were a part of the industrial automation sector became a part of this.

2.3.1 Classic OPC

The OPC foundation in the recent years has defined a number of software interfaces to standardize the information flow, right from the process level to the management level. The main use cases for the use of OPC are interfaces for industrial automation applications like HMIs and SCADA systems. They consume data from devices and also provide data on current and historical events for management applications. Three major OPC specifications have been developed for different applications within the industrial

applications. They are: Data Access (DA), Alarm & Events (A&E), and Historical Data Access (HDA). The DA specification describes access to current process data, A&E describes an interface for event-based information, including acknowledgement of process alarms, and HDA describes functions to access archived data.

Information is the core of the data transfer and OPC handles this by using a client server approach. An OPC server encapsulates the source of process information like a device and makes the information available via its interface. An OPC client, on the other hand connects to the OPC server and can access and consume the offered data. The applications that make use of this service can be both a client and a server.

.

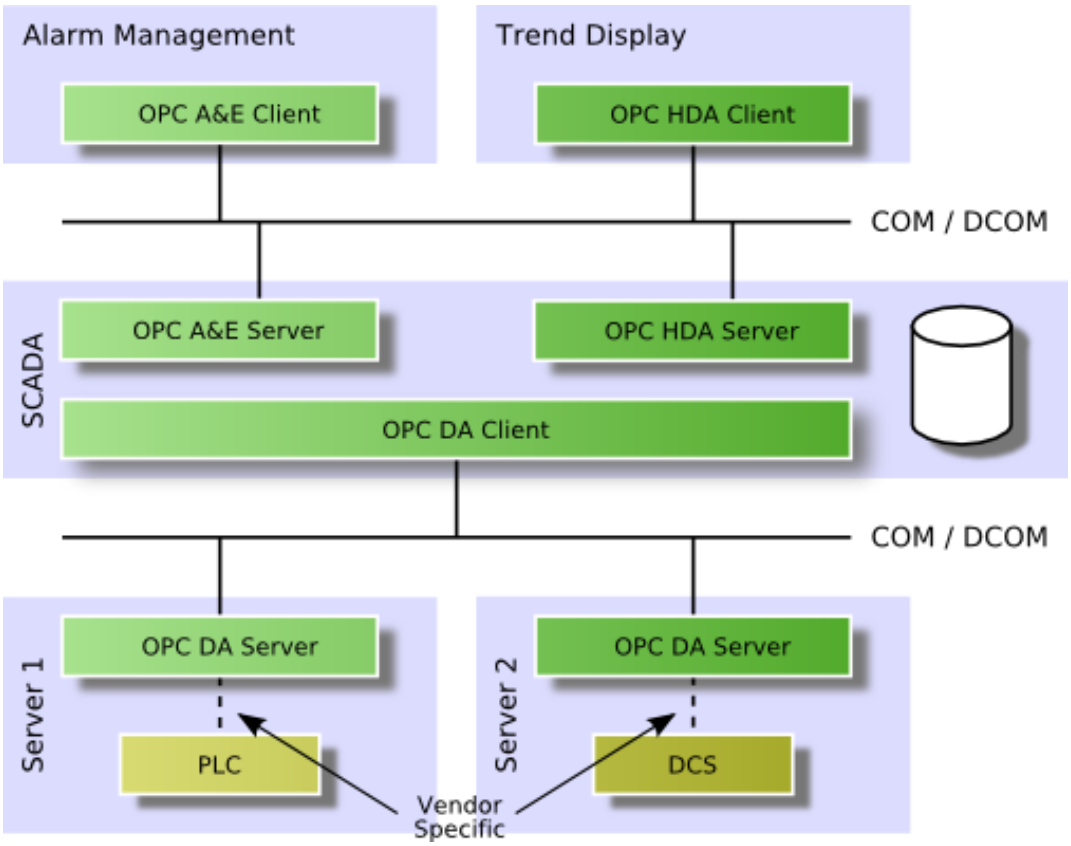

Figure 2.2: OPC Client and Servers with COM and DCOM technology from Microsoft.

The above figure 2.2, shows the connection between the client and servers used within a classic Microsoft communication service namely COM and DCOM. The positives of using this approach is that the specification work could be reduced in regard to the different definitions for APIs. The requirement also defines a network protocol or a mechanism for inter-process communication. COM and DCOM provide a transparent

mechanism for a client to call methods on a COM-object. The COM-object can be in a server running in the same process, in another process, or on another network node. Using this system reduced the development time and also the time-to-market of the OPC. There are two main negatives are the Windows-platform-dependency of OPC and the DCOM issues when using remote communication with OPC. This is primarily because the DCOM is very difficult to configure in general, cannot be used for internet communication in general and moreover, has very long and non-configurable timeouts. The following section shows a brief description of the three main specifications.

1. **Data Access** - The OPC Data Access interface enables reading, writing, and also monitoring of variables containing current process data. The main use case is to move real-time data from PLCs and other control devices to HMIs and other display clients. This is considered as the most important specification. The DA clients explicitly select the variables to perform read, write or to monitor in the server. The client establishes a connection to the server by creating an OPC Server object. The server also offers different methods to navigate through the address space and the hierarchy. It helps find items and their properties like data type and access rights.

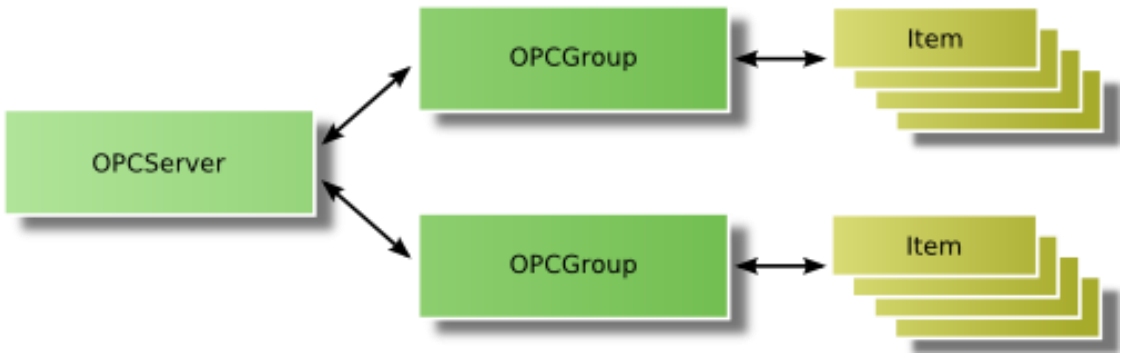

Figure 2.3: OPC Client access to the Server with objects

The client accesses the data on the server by the help of different items. The items created are grouped into different groups through with a connection request can be made to the server [Figure 2.3]. When added to a group, items can be read or written by the client. However, the preferred way for the cyclic reading of data by the client is monitoring the value changes in the server. The client also periodically updates the data of interest depending on the application. OPC

provides real-time data that may not permanently be accessible, for example, when the communication to a device gets temporarily interrupted. The Classic OPC technology handles this issue by providing timestamp and quality for the delivered data. There are three qualities specified for data accuracy namely, accurate (good), not available (bad), or unknown (uncertain).

2. **Alarms and Events** – the alarms and events interface enables the reception of event notifications and alarm notifications. Alarms are nothing but notifications that inform the client about the change of a condition in the process. Whereas, Events are single notifications that inform the client that an event has occurred. To receive notifications, the OPC A&E client connects to the server, subscribes for notifications, and receives all notifications triggered in the server. The notifications can be limited by using filters as a criteria.

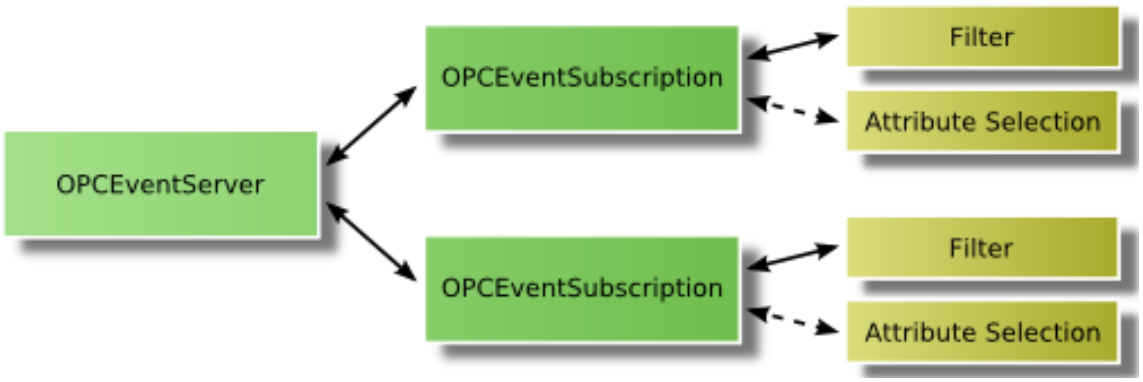

Figure 2.4: OPC Client access to the Server for Alarms and Events

The client connects to the server to communicate by using filters and selecting attributes. The OPC client connects by creating an OPCEventServer object in the A&E server in the first step and by generating an OPCEventSubscription used to receive the event messages in the second step [Figure 2.4]. In this specification there is no explicit request for information but the quantity of items can be specified. This is done by setting certain filter criteria, for example, filter by event types, by priority, or by event source.

3. **Data Historical Access** – The OPC Historical Data Access provides access to data already stored. The main area of use is the reading of historical data in three different ways. The first mechanism reads raw data from the archive, where the client defines one or more variables and the time domain he wants to read. The server returns all values archived for the specified time range up to the maximum number of values defined by the client. The values from one or more variable re read from the timestamps in the second mechanism. The third read mechanism computes aggregate values from data in the history database for the specified time domain for one or more variables. Values include always the associated quality and timestamp. Historical Data Access also provides functionalities for inserting, replacing, and deleting data in the history database.

The classic specifications working with each other are shown in the figure 2.5. The OPC Overview forms the base layer. It is also extended to OPC Common defining interfaces and behavior that is common to all COM-based OPC specifications. Next comes the OPC Security and specifies how to control client access to servers to protect sensitive information and to guard against unauthorized modification of process parameters. The transport values with complex structured data types are described in the lower layer consisting of Complex Data.

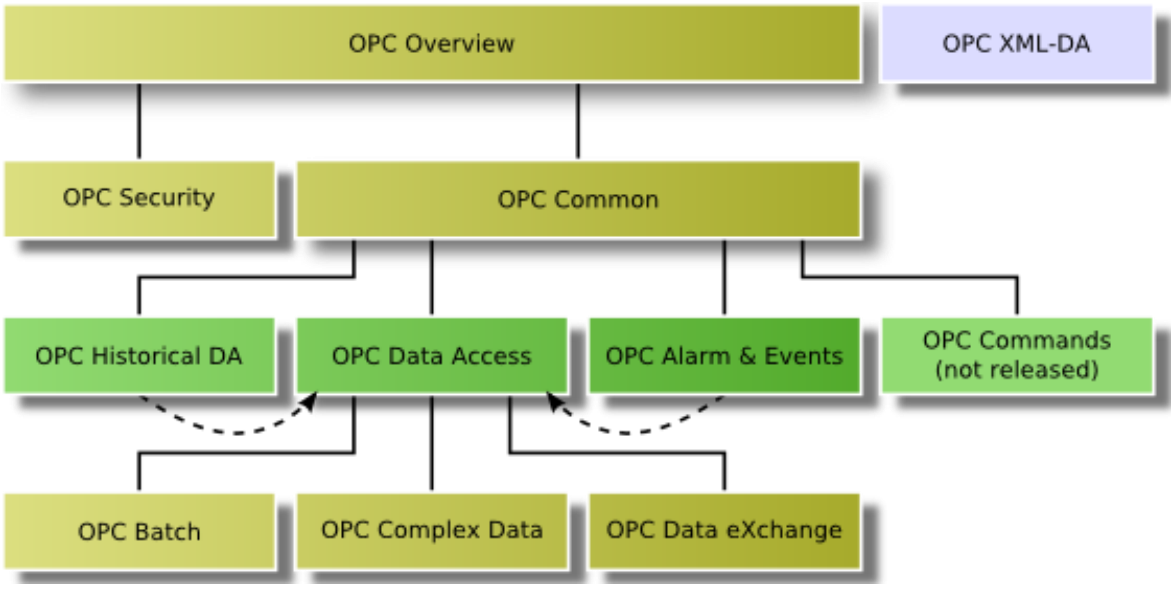

Figure 2.5: OPC Client access to the Server with objects

OPC Complex Data, OPC Batch, and OPC Data Exchange (DX) are extensions to OPC DA. OPC DX specifies the data exchange between Data Access servers. This is done by defining the client behavior and the configuration interfaces for the client inside a server. The OPC Batch process extends DA for the specialized needs of batch processes. It also provides information on exchange of equipment capabilities. The OPC Commands section had not yet been developed during the classic OPC specifications but was later implemented with OPC UA. All these layers make up the classic OPC specifications [19].

2.3.2 Motivation for OPC UA

This section describes the changes which led to the development of OPC UA from the classic OPC description. With the successful adoption of OPC in thousands of products, OPC is used today as standardized interface between automation systems in different levels of the automation pyramid. It is even used in a lot of areas where it was not designed for, and there are many more areas where manufacturers want to use a standard like OPC but are not able to use it because of the COM dependency of OPC or because of the limitations for remote access using DCOM. There are several reasons why just creating Web Service versions of the successful OPC specification did not cover the requirements for a new OPC generation. One reason was the poor performance of XML Web Service. This is in comparison with the original COM version. Besides, XML stacks also cause interoperability problems.

The OPC Unified Architecture was born out of the desire to create a true replacement for all existing COM-based specifications without losing any features or performance. It must also cover all requirements for platform-independent system interfaces with rich and extensible modeling capabilities. It is further concluded that the most important requirements of OPC UA are as follows,

- Communication between distributed systems
    - Robustness and Fault Tolerence
    - Reliability on redundancies
    - Platform-independence
    - Scalability
    - High Performance
    - Internet and Firewalls
    - Security and Access Control

    - Interoperability

- Modelling Data
    - Common model for all OPC data
    - Object-oriented
    - Extensible type system
    - Meta information
    - Complex data and methods
    - Scalability from simple to complex models
    - Abstract base model
    - Base for other standard data models

Classic OPC was designed as device driver interface. OPC is used as system interface today; therefore, the reliability for the communication between distributed systems is very important. Robustness and fault-tolerance are the important requirement, also including redundancy for high availability. Platform-independence and scalability is necessary to be able to integrate OPC interfaces directly into the systems running on many different platforms. To replace proprietary communication, an important requirement is always high-performance in intranet environments. This must also include internet communications through firewalls with the option of adding security layers and maintain access controls.

Modeling of data was very limited in Classic OPC and needed to be enhanced by providing a common, object-oriented model for all OPC data. This model must include an extensible type system to be able to offer Meta information and to describe also complex systems. The availability of methods provided and described by servers and callable by clients is a powerful feature needed to make OPC flexible and extensible. This requires complex data to support the transportation of data. For this reason it is required to have a simple and abstract but extensible base model to be able to scale from simple to complex models.

Another important design goal was to allow an easy migration to OPC Unified Architecture to protect the investment in the very successful Classic OPC standards and to build upon the large installed base of OPC. These are the main motivators behind the creation of OPC UA [19].

2.3.3 Introduction to OPC UA

To reach the defined goals from the previous model, the OPC UA complies with an architecture built on different layers as shown in figure 2.6.

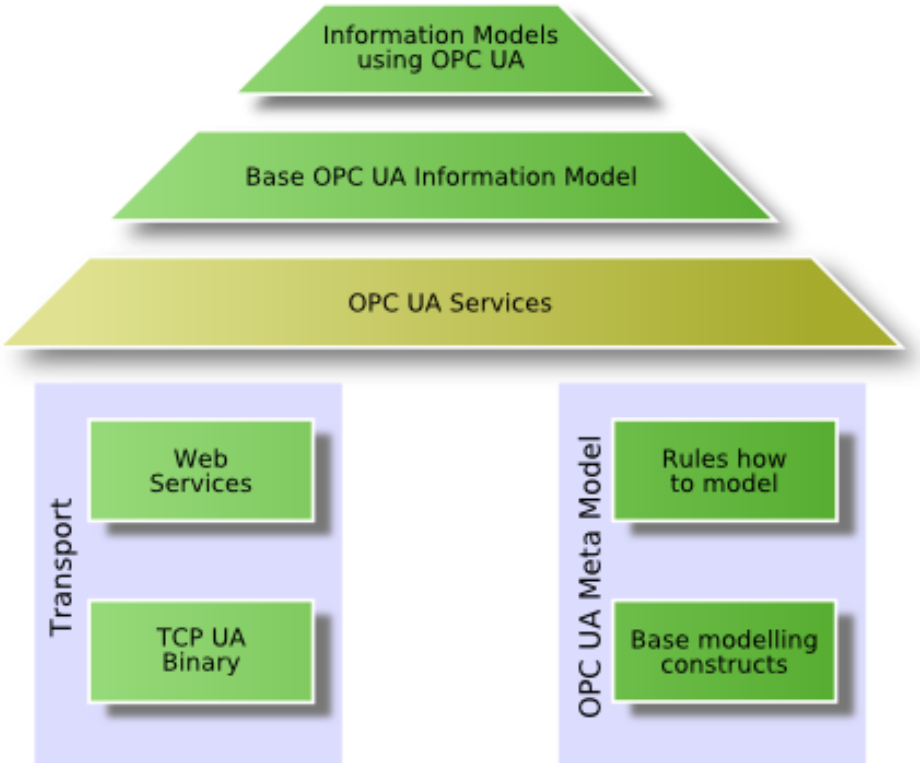

Figure 2.6: Architectural layers of OPC UA

There are five different layers which form the architectural definition. The most important part is the transport layer. This layer defines the different mechanisms optimized for different use cases. The first version of OPC UA is defining an optimized binary TCP protocol for high performance intranet communication as well as a mapping to accepted internet standards like Web Services, XML, and HTTP for firewall-friendly internet communication. Both transports are using the same message-based security model known from Web Services. The abstract communication model does not depend on a specific protocol mapping and allows adding new protocols in the future.

The data modeling defines the rules and base building blocks necessary to expose an information model with OPC UA. It defines also the entry points into the address space and base types used to build a type hierarchy. This base can be extended by information

models building on top of the abstract modeling concepts. In addition, it defines some enhanced concepts like describing state machines used in different information models. The next layer is the UA Services layer. This layer defines the interface between servers as supplier of an information model and clients as consumers of that information model. The Services are defined in an abstract manner. They are using the transport mechanisms to exchange the data between client and server.

The information models defined by OPC vendors and parties involved are shown in the figure 2.7. This basic concept of OPC UA enables an OPC UA client to access the smallest pieces of data without the need to understand the whole model exposed by complex systems. OPC UA clients also understanding specific models can use more enhanced features defined for special domains and use cases.

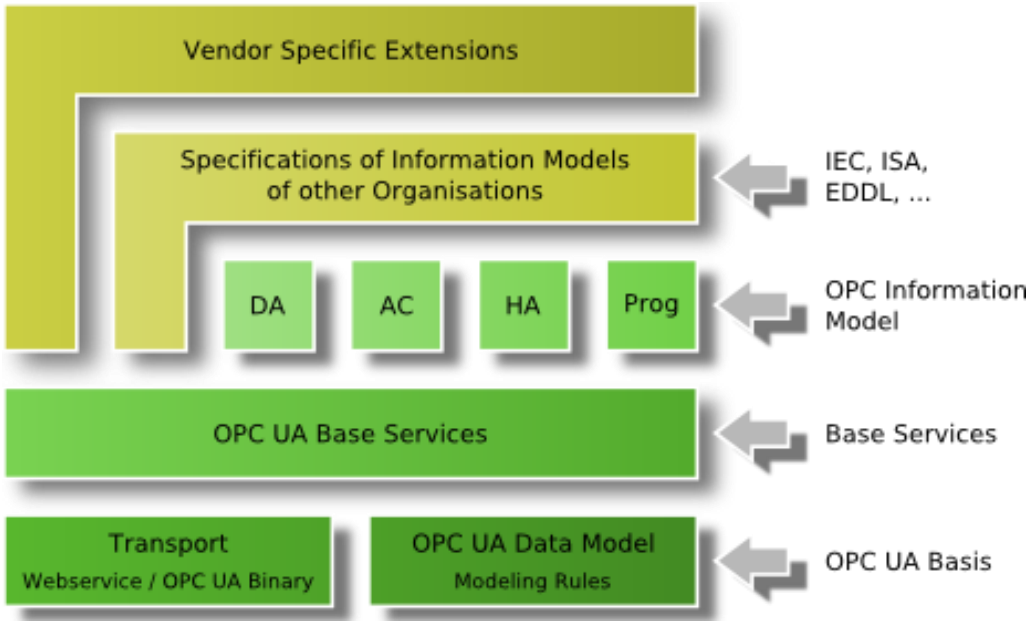

Figure 2.7: Information Model of OPC UA

The information models for the domain of process information are defined by OPC UA on top of the base specifications. DA defines automation-data-specific extensions such as the modeling of analog or discrete data and how to expose quality of service. All other DA features are already covered by the base. Alarm & Conditions (AC) specifies an advanced model for process alarm management and condition monitoring. Historical Access (HA) defines the mechanisms to access historical data and historical events.

Programs also specifies a mechanism to start, manipulate, and monitor the execution of programs. This allows other organizations and programs operators to build their models on top of the UA base or on top of the OPC information model. Some of the examples for standards already working on mappings to OPC UA are Field Device Integration (FDI) combining Electronic Device Description Language (EDDL), and Field Device Tool (FDT). FDT and EDDL are used to describe, to configure, and to monitor devices and PLCopen, a standard for PLC programming languages [19].

2.3.4 OPC UA Specifications

This section describes the various specifications and different parts that are partitioned as per the IEC standardization. OPC UA will be known as IEC 62541 standards. The figure 2.8 shows an overview of all specification parts split into the core specifications defining the base for OPC UA and the access type specific parts mainly specifying the OPC UA information models.

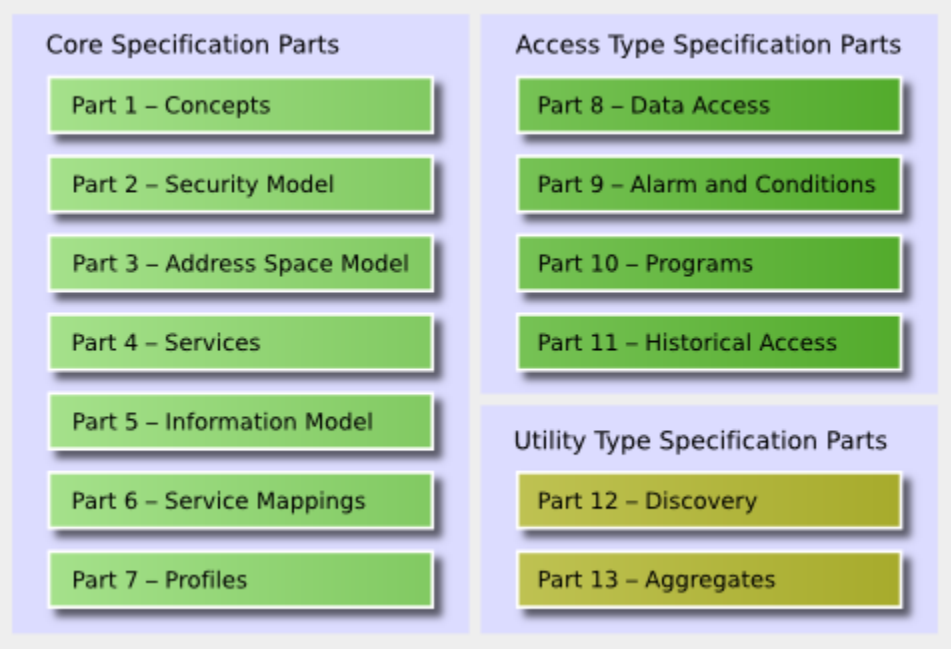

Figure 2.8: OPC UA Specifications

The first two layers are not normative. This means that they are not bound to any standard and are open for the end user to change/modify depending on the needs of the application. The concepts part UA Part 1 gives an overview about OPC UA and UA Part

2 describes the security requirements and the security model for OPC UA. The real advantages of OPC UA really comes into the picture here as various security models can be built on the OPC UA to make a system more robust and secure. Most important to understand how to model and access information are part 3 and 4. These two specifications are the key documents for the design and development of OPC UA applications. The Address Space model contains specifications for entities like access levels, arrays, data items, static array variables, variables of the type Boolean, byte, string, double along with images and also XML data. This section also describes the usage of the nodes and objects. It specifies the building blocks to expose instance and type information and thus the OPC UA Meta model used to describe and expose information models and to build an OPC UA server address space.

The abstract UA Services defined in UA Part 4 represent the possible interactions between UA client and UA server applications. The client uses the Services to find and access information provided by the server. The Services are abstract because they are defining the information to be exchanged between UA applications but not the concrete representation on the wire and also not the concrete representation in an API used by the applications. Figure 2.9 shows the layered communication architecture of OPC UA.

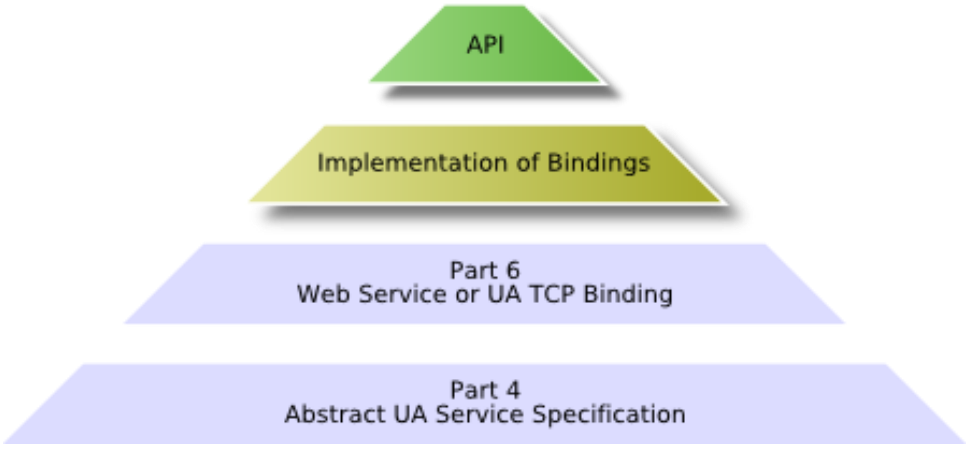

Figure 2.9: Layered communication architecture of OPC UA

The mapping of the UA Services to messages, the security mechanisms applied to the messages, and the concrete wire transport of the messages are defined in UA Part 6. Only implementers of UA stacks need to completely understand this specification. Since the OPC Foundation supplies proper UA stacks, typical UA application architects and programmers do not need to read this specification.

The base information model specified in UA Part 5 provides the framework for all information models using OPC UA. It defines the following:

- The entry points into the address space used by clients to navigate through the instances and types of an OPC UA server
- The base types building the root for the different type hierarchies
- The built-in but extensible types like object types and data types
- The Server Object providing capability and diagnostic information.

The profiles are defining useful subsets of OPC UA features in UA Part 7. Such a subset must be implemented completely by an UA application to ensure interoperability for the defined subset. The specification defines the subsets on two levels. The first level are Conformance Units defining a small set of functionality that is always used together and can be tested with Compliance Test Tools and verified as unit. The second level are Profiles composed of a list of conformance units. A profile must be implemented completely and will be verified as complete set during the certification of OPC UA products. The list of supported and used Profiles is exchanged during the connection establishment between client and server and allows the applications to determine if the needed features are supported by the communication partner.

Part 8 is the Data Access information model that defines how to represent and use automation data and specific characteristics like engineering units. Part 9 pertains to the alarms and conditions information model specifies process alarm and condition monitoring specific state machines and types of events. The Programs information model defines a base state machine for the execution, manipulation, and monitoring of programs in UA Part 10.

Another important feature of OPC UA is the ability to access historical data. The Historical Access information model in UA Part 11 specifies the use of the history access Services and how to present information about the configuration of data and event history. The aggregates used to compute aggregated values from raw data samples are specified in UA Part 13. The aggregates are used for historical access as well as the monitoring of current values.

The final part defines the methods in which a client and a server can be discovered. This section defines how to find servers in the network and how a client can get the necessary information to be able to establish a connection to a certain server [19].

2.3.5 Software Layers

OPC UA uses a similar client–server concept like Classic OPC. An application that wants to expose its own information to other applications is called UA server and an application that wants to consume information from other applications is called UA client. But it is expected that much more applications will be both UA server and UA client in one application than in Classic OPC. One reason is that more UA servers will be integrated directly in devices. Implementing also a UA client enables device to device communication. Another reason is the use of OPC UA as configuration interface, where UA clients are also UA servers to be configured via OPC UA.

A typical OPC UA application is composed of three software layers shown in the following figure 2.10. The complete software stack can be implemented with C/C++, .NET, or JAVA. OPC UA is not limited to these programming languages and development platforms, but only these environments are currently used for implementing the OPC Foundation UA Stack deliverables.

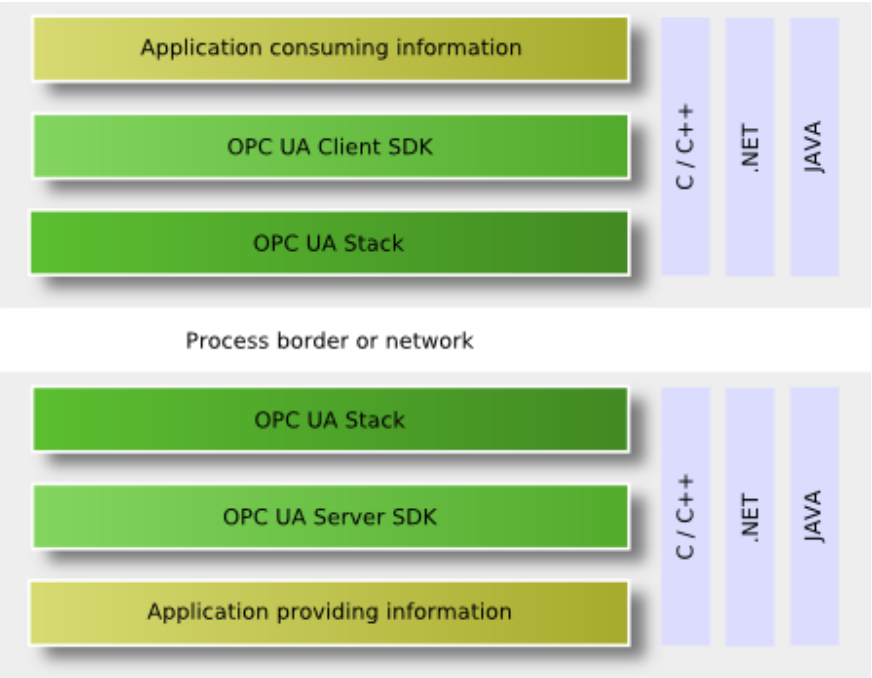

Figure 2.10: Software Layers of OPC UA

An OPC UA Application is a system that wants to expose or to consume data via OPC UA. It contains the specific functionality for the application and the mapping of this functionality to OPC UA by using an OPC UA Stack and an OPC UA Software

Development Kit (SDK). The client server SDK also implements common OPC UA functionality that is part of the application layer, since the UA Stacks implement only the communication channels. An OPC UA SDK reduces the development effort and facilitates faster interoperability for an OPC UA application.

An OPC UA Stack implements the different OPC UA transport mappings defined in UA Part 6 shown in figure 2.11. The Stack is used to invoke UA Services across process or network boundaries. OPC UA defines three Stack layers and different profiles for each layer. The message encoding layer defines the serialization of Service parameters in a binary and a XML format. This is followed by the message security layer which specifies how the messages must be secured by using the Web Service security standards or a UA binary version of the Web Service standards. The message transport layer defines the used network protocol, which could be UA TCP or HTTP and SOAP for Web Services. The following figure illustrates the different UA communication stack layers. The implementation of the layers in a UA Stack and the resulting API for the applications is not part of the OPC UA specification. These stacks are language dependent APIs for UA client and UA server applications, but the Services and their parameters are similar and based on the abstract Service definition in UA Part 4.

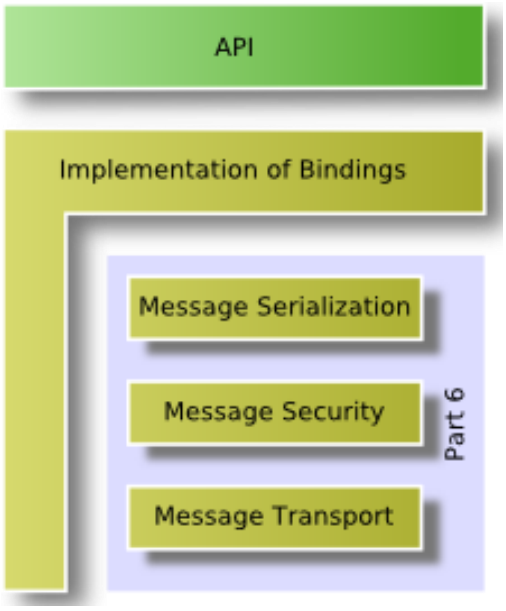

Figure 2.11: Transport Mappings of OPC UA

OPC UA is much more flexible and has much more features than all Classic OPC specifications together, but it incorporates all successful concepts of existing OPC specification, fixes known issues in existing standards, and adds standardization for a lot

of additional use cases. It was an important design goal to allow an easy migration from Classic OPC to OPC UA. For this reason most features known from Classic OPC can be found in OPC UA using sometimes slightly different terminology. It is not possible to expose all UA features with Classic OPC interfaces, but it is no problem to map Classic OPC features to OPC UA. OPC UA allows a simple mapping and offers migration strategies to integrate OPC products based on previous OPC standards. One part of the migration strategy does not even require a change in existing products. Wrappers and proxies provided by the OPC Foundation are able to translate the different Classic OPC interfaces into OPC UA and vice-versa. This first level in the migration strategy can be used by vendors to support OPC UA for legacy products.

The second level of migration to OPC UA is the integration of OPC UA directly into existing products without adding OPC UA specific features. This step does not require changes in interfaces used between systems and their OPC communication components today. It is much easier to integrate new components if the existing interfaces of a system do not have to be changed. The advantage over the use of wrappers or proxies is a better performance and less configuration and engineering efforts by removing an additional software layer. The direct integration will make it easier to remove potential limitations of wrappers and proxies and allows an iterative development approach by adding OPC UA features step by step.

The third level may require changing internal interfaces of the product to support all features of OPC UA that are of interest for the product. OPC UA will allow systems to make product features available in a standard way, which they cannot expose today without the new options provided by OPC UA. For end users, it is important to have the wrappers and proxies available to migrate the large installed base of Classic OPC products to OPC UA. End users can install wrappers and proxies for tunneling Classic OPC through firewalls, including secure transmission over the internet and authenticated access, so they simply add value to existing industry proven solutions.

The very powerful concepts for extending this base enables developers to expose more and more of their system features via OPC UA. This leads to an iterative development and improvement process [19].

2.4 Server SDK

The OPC UA consists of a Software Development Kit. The SDK simplifies the UA stack APIs, implements common UA functionality needed in most or all UA applications, provides base functionality and helper functions, implements the security handling and

provides samples for common use cases. The main criteria here is to provide a standard interface for all vendor specific systems in a standard way.

To make the implementation of JAVA based OPC UA Servers as easy as possible, the SDK does the following,

- Implements all common UA functionality as reference implementation
- Defines interfaces to integrate the vendor system data into the OPC UA Server
- Provides main and related classes implementing often used functionality for a vendor system integration
- Provides wrapper classes for all system functionality and OPC UA structures
- Provides UA stack platform layers for Windows and Linux

As additional feature the SDK itself is platform independent. The following section is a depiction of the main components involved in defining the server SDK which leads to the final integration of the server application. This is applicable to the other languages such as C++ as well. Figure 2.12 shoes the main modules involved in defining a server application.

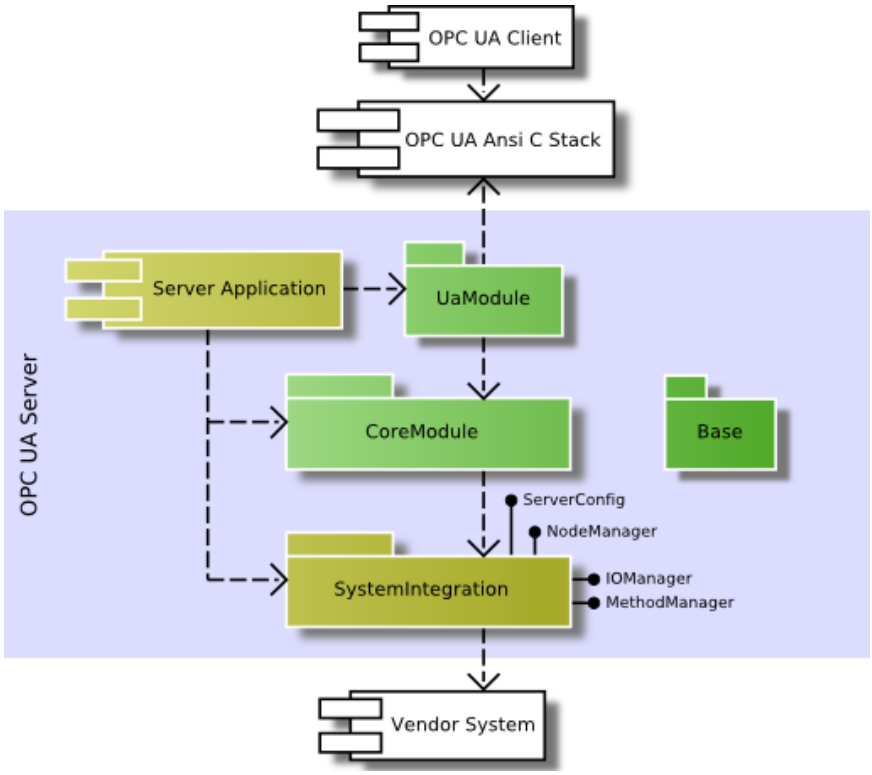

Figure 2.12: Main modules of server application

UA functionality that is common for all UA Servers is implemented in the following section. It is not necessary to know the internal structure of these modules and the user of the SDK needs to know only the interfaces and helper classes provided by the modules. The following are the individual components of the server SDK.

1. **Base** - The Base Module implements the base functionality necessary for the OPC UA Client and Server applications. The JAVA classes in the Base Module are using the UA structures and the platform layer defined by the OPC UA ANSI C Stack. The Base Module implements its own platform layer. This means that for all system specific functionality that is not provided by the UA Stack like functions to rename files or for trace functionality. The platform layer however, has an additional functionality in which it allows changing of the functionality based on the product requirements like the trace functionality. The main task of this layer however, is to provide implementations for different platforms.

2. **Core Module** - The Core Module manages the address space of the OPC UA Server. It also implements the base functionality of a Server, which means that it performs tasks like session management and also the OPC Foundation defined part of the address space. The Core Module defines also the interfaces used by the implementer of the OPC UA Server. This helps integrate the vendor specific system with the UA Server SDK. It also helps in managing the access to the vendor system integration. The structure of the address space and the functionality is optimized for OPC UA but the core module can also provide its data via other interfaces like COM Data Access or XML Data Access.

3. **UA Module** - The UA Module implements the functionality necessary for OPC UA communication like Subscription handling and the UA Service implementations. The implementation of this module uses the Core Module to access the data provided by the OPC UA Server. The UA Module is using the OPC UA Ansi C Stack for communication with the OPC UA Client.

4. **Server Application** – the advantage of the OPC UA server is that it can be integrated into an existing application or better, can run as standalone application. The integration is trivial and requires only loading the core module and the UA module during startup of the application. This helps in unloading the modules during shut down of the application.

5. **System Integration** - UA functionality that is specific to the product is implemented in the System Integration part of the OPC UA Server. The SDK defines interfaces like the required interfaces. These interfaces are ServerConfig, NodeManager and IOManager. Additional to this there are optional interfaces such as MethodManager, EventManager and HistoryManager to plug in the integration of the vendor system into the Core Module. The OPC UA Server can be a standalone application or can be part of an existing application. In both cases the application must load and start the Core Module, the System Integration and the UA Module during startup of the application and must shut down and unload these modules during shut down of the application. The stack provides APIs for the Client and Server applications and hides all the message transport functionality [19].

2.5 Client SDK

The OPC UA Client SDK defines APIs which the Application Developer can use to communicate with an OPC UA Server. There are numerous helper classes which store different types of information. However, there are three main classes with which the application developer works with, these are listed as follows,

1. **UaDiscovery Class** - The UaDiscovery class provides access to the discovery endpoint of either the Local Discovery Server (LDS) or a Server. It allows applications to discover servers and find the available endpoints. The following Figure 2.13 shows how the client and server make use of the Discovery servers to connect to the endpoints.

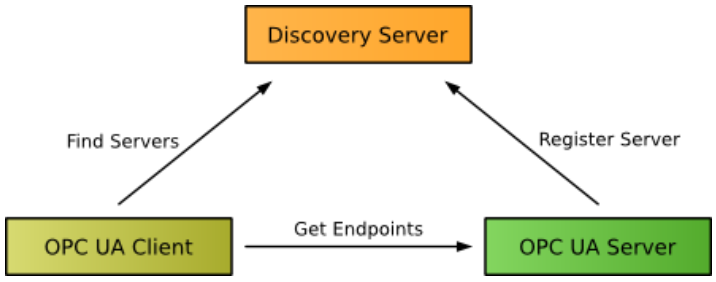

Figure 2.13: Discovery server in OPC UA

2. **UaSession Class** - A UaSession represents a connection with a single Server. It maintains a list of active Subscriptions. It monitors the connection status, handles automatic reconnects and informs the application about connection status changes. A process can create many Session objects connected to different Servers. The Session object is responsible for monitoring the connection with the Server and attempting to reconnect if there are any errors. The application can receive notifications of changes to the connection status through the callback

3. **UaSubscription Class** - A UaSubscription represents an active Subscription with a Server. A Subscription is owned by one Session and will have one or more MonitoredItems. MonitoredItems are used to monitor data or events produced by individual nodes in the Server address space. The data and event streams created with monitored items are provided as callbacks through a callback technique. It does not maintain a local copy of the parameters and MonitoredItems. If the Subscription is no longer valid, the user of the SDK is responsible for re-creating the Subscription on the Server.

In summary, the different types of C2X standards have been explicitly explained. The standards and protocols were then specified in detail. This was followed by a description of OPC history and how the new OPC UA was introduced as the standard for industrial automation. The motivation and the familiarization of the OPC UA was then carried over to the various specifications and software layers involved in the architecture. The section was then concluded by defining the server SDK and the client SDK in brief to mention about the connectivity and usage in this thesis. The next section throws light on the existing technologies at DLR and their limitations [19].

3 Current Techniques at DLR

3.1 Introduction

This section introduces the current techniques that are already in place and being employed at DLR. In the beginning, the traffic light algorithm used universally is explained which is followed by the hardware architecture. It is followed by the problem exposition and a proposal for the main idea behind the thesis.

3.2 Traffic Light Algorithm

The traffic light algorithm uses an interesting algorithm which determines the state of the signal. As described in [18], each traffic light group at an intersection tries to minimise the waiting times in front of its traffic lights. For that purpose some parameters have to be introduced.

A traffic light is assigned a red phase t_r and a green phase t_g , which means that the proportions for the horizontal street and the vertical street, respectively and vice versa for the traffic light controlling the orthogonal direction. This causes a total life cycle of time equal to $t_{cycle} = t_r + t_g$. Depending on the length of the traffic the signals are stretched in time or shortened. The following flowchart [Figure 3.1] shows the behavior of each agent based traffic signal.

For the intelligence of the traffic lights the first important parameter is the “looking distance” d_{look} of each traffic light, which is set to 10 cells (approx. 75 meter). The second parameter is the time interval t_{decide} in which a traffic light probably makes a decision (increasing or decreasing the green time for the north/east-bound direction by one second). This parameter is set to the triple of t_{cycle} throughout, so a decision is performed approximately every 5 minutes. As shown in the flowchart, each traffic light counts the vehicles waiting in line within its looking distance during a red phase. If the ratio of vehicles waiting compared to the others is positive and greater than a certain value, then the green signal duration is increased. If the value is negative or less than a certain value then the duration is decreased.

This is how the traffic algorithm works in the real world scenario. The same is being considered at the traffic department of DLR as well.

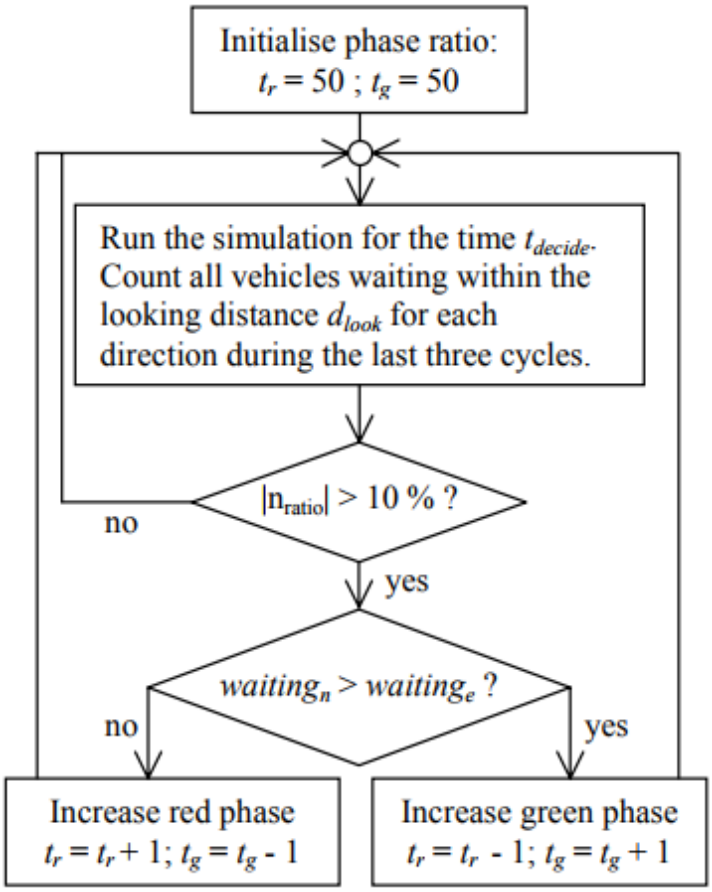

Figure 3.1: Flowchart representing the behavior of each agent-based traffic signal

3.3 Dominion

At the Institute of Transportation Systems of the DLR, DOMINION will be involved Framework developed as the basis for the realization of advanced Driver assistance systems are used [GHHK08]. DOMINION is based on the service-oriented architecture (SOA) approach and pursues a user-centered development approach. Since features and concepts often change in the short term, flexible functional development is in the foreground. The DOMINION framework allows for the design and continuous evaluation of technical prototypes at various stages of the development process. This includes both simulations and field studies in real traffic. Within the DOMINION software architecture, software modules are mapped as elements of a distributed system that provides specific services for different domains [GHH08]. A domain master manages the services and ensures access to one common data core.

The DOMINION framework covers the entire range of requirements in the development and evaluation system in the context of advanced driver assistance systems. In order to ensure a high flexibility with regard to changes during the process from the conception through the implementation and testing of prototypes in simulations as well as field tests, a separation of functional code and the formal service description of specification is performed. The DOMINION framework uses the definition of a service to provide the basic structure for the implementation and ultimately controls the implementation execution. The DOMINION core provides common data structures centrally. The implementation of the processing and the management of the core data structures is provided for the service as a kind of black box. For example, the periodic execution of a process is governed directly by the formal specification of the associated service. This has the disadvantage that, for example, within the implementation of a service for sending messages in a fixed interval, it is not possible to influence the execution with regard to compliance with the transmission frequency. Furthermore, estimating the exact duration of a service is difficult.

As for the investigations under this work no simulation is necessary but field tests are conducted exclusively on existing infrastructure. A new development offers the advantage that the special requirements can be focused in the context considered here. The limitations resulting from the black box functionality of the DOMINION framework are avoided, and aspects such as the subjectivity of a particular send interval can be considered. In the conception, previous experiences or results from the application of DOMINION can be included. Furthermore, a subsequent comparison to an implementation using the DOMINION framework is possible. A standalone software for measuring and another for evaluating and visualizing the measurement results allows specific extensions for other application cases or special features of deviating C2X hardware and software. With a homogeneous output format, it is subsequently possible to uniformly process and display the measurement results with an unaltered analysis tool. In contrast to an implementation as a closed measurement and analysis tool, the functionality of the measurement tool can thus be adapted to the aspects to be examined and the C2X hardware and software used. In addition, the processing of the analysis tool can be adapted as needed without affecting the functionality of the measurement tool [17].

3.4 Thesis Problem Statement and Proposal

The underlying functionality of the Road Traffic Infrastructure is modeled as depicted in the previous sections. The application unit processes the signals and wraps it in a message format which is readable by the applications at the receiving end. Some of the message formats are SPaT, DENM, CAM messages.

However, it must be noted that the messages have no robust security layer wrapped around them. This means that the messages are not encrypted and they are vulnerable to potential hacks or attacks. This can pose to be a serious threat to the entire infrastructure in the future since the whole system can be compromised.

My thesis at DLR attempts to address this issue. While researching for possible solutions to address the issue of security, one topic was come across. This possibility was the usage of a server client architecture to send and receive messages and vice versa. Both the server and the client are interoperable which means both can either send or receive data as per the needs of the application. This communication technique is widely used in the automation industry but surprising not in the automotive industry. OPC UA uses OLE for process control. This means that it is built on the method of object linking in the embedded level. This helps build a robust application which is scalable and secure while being robust at the same time.

OPC finds its usage largely in C2X communication and even M2M communication since it is basically an interpreter between machines. It is for the same reason the automation industry uses it in such abundance. However, the biggest advantage of this system is within the underlying security feature. The OPC UA provides amongst others, its own layer of security which can be used to our advantage in the communication between the mobile road side unit and the cars or any other machine. In addition to this, it leaves an option of adding an own layer of security which is of the developer's choice.

It is hence proposed in this thesis to explore the option of incorporating the OPC UA server client architecture to bring security to the already existing infrastructure at the Road and Traffic Department at DLR. In other words, the feasibility of OPC UA should be researched and examined for the purpose of the future of communication in Car-2X communication.

4 Concept Development

This section depicts the concept that is being proposed in contention with the existing architecture. A proposal to the question will be laid out and appropriate techniques to tackle these topics will be discussed as well.

4.1 Existing Architecture

The existing hardware setup of the ITS Roadside Station (IRS) is shown in Figure 4.1. Each IRS however consists of the following components [12],

- A Communication Control Unit (CCU), a specialized communication device for C2X communication according to IEEE 802.11p
- An Application Unit (AU), a computer to process applications requiring a higher computing capacity in the field. Furthermore a database can store process data (e.g. sensor input).
- A GPS receiver for both time and positioning reference
- An interface to the adjacent traffic light controller
- A connection to the backbone network

Additionally, a specific IRS can be equipped with one or more of the following components (Dotted line in Figure 4.1):

- A connection to the traffic tower (e.g. via ethernet)
- Communication modules such as regular Wireless LAN (WLAN)
- Sensors for traffic detection such as radar sensors or cameras

Modularity has been a major paradigm in the development of the IRS architecture. Inside the IRS the different components are interlinked by Ethernet LAN. This means that any component could be easily replaced by another one. Because of this modular architecture the same setup of AU, CCU, GPS receiver and additional sensors is also used inside the vehicle as an ITS Vehicle Station (IVS).

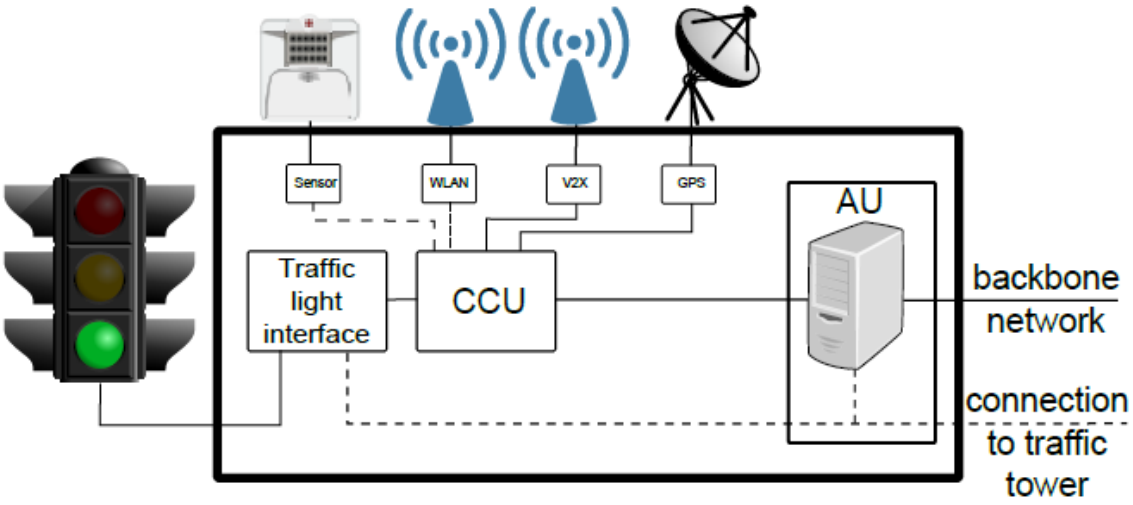

Figure 4.1: ITS Roadside Station Hardware Setup

The Communication Control Unit (CCU) is a device that handles the Vehicle2X communication. It periodically sends beacon frames on the V2X band and it transmits messages whenever requested by the Application Unit (AU). Any application-specific software runs on the AU, which is an industrial-type computer with some hardware modifications. In the current setup, the AU software receives signal state messages that are sent by the traffic light interface on a UDP port and forwards these messages to the CCU without changing them. In future applications the AU will compose new messages on demand and send them to the CCU for transmitting via V2X.

The GPS receiver is used to add accurate time stamps to the V2X messages. The precisely measured position is stored inside the CCU and transmitted with everybeacon frame.

In order to get information from the traffic light controller, each IRS has a specialized interface that allows read only communication between the traffic light controller and the AU. Both the traffic light controller and the interface were developed by Siemens in order to guarantee safe operation of the traffic light system.

The backbone network connects the IRS with each other and with the ITS Central Station, a backend server which is currently being specified. There are three feasible ways to connect the IRS backbone a UMTS mobile radio connection, a regular Wireless LAN or an SDSL connection. The other IRS locations will be connected via a regular Wireless LAN which serves two purposes: interlinking IRS that do not have an own uplink to the backbone network and providing information to mobile devices of road users and non-road users via IEEE 802.11g. A special WLAN network is being installed that uses mesh technology to realize this task. A mesh network provides roaming between the

various WLAN access points, so a mobile device can move inside the mesh network without losing connection to the network. On the one hand, the IRS provides static information on the intersection topology, the lanes on the intersection and the allowed maneuvers on the lanes (map message). On the other hand, dynamic information on the traffic light is communicated to the vehicle. The SPAT message contains information on all the signals of the traffic light for any (relevant) direction and traffic types [15].

4.2 Proposal of a New Architecture

The new proposal for an alternate infrastructure is as shown in Figure 4.2. The major changes to the already existing model are the introduction of the OPC server and client which will be explained further on.

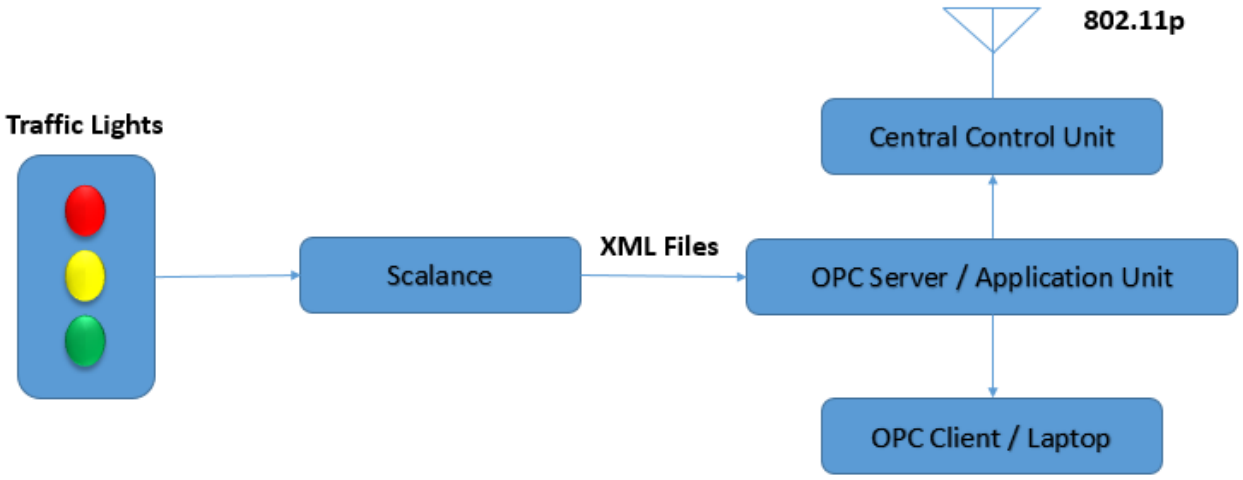

Figure 4.2: Proposal for Implementation

As seen in the previous architecture, the traffic lights is used as a read only source in order to evaluate the signals and use it in a secure connection. These raw signals that are generated are then sent to Scalance for routing. The output of the Scalance is the beginning point for the description of a new infrastructure. The key points can also be encapsulated as follows,

- The mobile traffic light feeds raw messages in bytes to the Scalance. This is a read only event and the signals are sent periodically until the exit condition is

met. The messages are fed into an XML file dump which contains all the data regarding the states of the signal

- The Scalance is used for routing these signals securely to the underlying infrastructure unit which is the application unit

- The application unit is the point of the infrastructure which is vulnerable to external attacks at the moment since it lacks security. This is where the OPC UA comes into play

- The OPC Server is installed in the application unit which receives the xml data which consists of SPaT messages. The OPC server adds to this message its own layer of security and makes the messages free from hacks. The server can also be set to send and receive events and subscriptions if needed. Here two types of security layers can be employed. One is the provided by the OPC UA standard and the other is open to the developer to implement his/her own layer of security if needed.

- The messages are now sent securely to the OPC UA client who can read the messages securely once the certificates are trusted.

- These messages can also be sent over to the CCU for a broadcast or to send information to another Infrastructure point. Hence, enabling a true M2M communication as well.

- These connections can be realized as an alternate to the existing infrastructure at DLR.

4.3 UA Expert

The UaExpert is a full-featured OPC UA Client demonstrating the capabilities of our C++ OPC UA Client SDK/Toolkit. The UaExpert is designed as a general purpose test client supporting OPC UA features like Data Access, Alarms & Conditions, Historical Access and calling of UA Methods. The UaExpert is a cross-platform OPC UA test client

programmed in C++ [16]. There are four main parts as shown in Figure, to the application and they are,

1. **Data Access View** - This plugin is shown in the center pane of UaExpert by default. You can (multi-) select UA Nodes in the Address Space window and drag-and-drop them into the DA View. The DA View creates a subscription and monitors the Nodes. Sampling rate and subscription interval can be changed by right-click into the DA View. The DA View was designed to show the classic view on OPC Servers solely concentrating on item monitoring and displaying values, timestamp and status of individual nodes.

2. **Alarm and Conditions View** - The Event View document can be added using the Add Document button in the menu bar. The Event plugin will be displayed in the center pane and consists of three major groups, the Configuration, the Event/Alarm view and the Details view showing the detailed information of an individually selected alarm.

3. **Historical Trend View** - The Historical plugin will be displayed in the center pane and consists of two major groups, the Configuration and the historical data view showing the values in a graphical trend view related to the requested time frame. The Historical Trend View supports two modes for fetching the data from the UA Server, the Single Update and the Cyclic Update.

4. **Performance View** - The performance plugin will be displayed in the center pane and consists of three major groups, the Configuration, the list of used nodes and the results showing the measurement in a graphical view. All nodes must be from the same UA Server and should have the same data type for easier interpretation of the results. The number of cycles gives you the number of calls performed for each measurement. The UaExpert will call the UA Service and measure the duration of each call. Alternatively you can choose the Duration option instead. Here the UaExpert will call the UA service as fast as possible in that time span and count how many calls could be performed.

4.4 Summary

In this chapter we have seen the existing limitations in the infrastructure. The limitations are solvable ad hence a real world solution is proposed. The usage of OPC UA can be a boon for the communication techniques used. A concept proposal is also laid out here.

The concept proposal takes into account the current techniques in detail. The architecture is depicted and the areas of improvement are picked upon for development. The OPC UA in summary has very good advantages not only when single connection requests are made to the server, but also when multiple connections have to be made. This interoperability between machines that use different languages can be a great advantage for any communication model that lays heavy emphasis on a secure transfer of data.

The following chapters show how the server client implementation can be made along with the use of third party applications that are built to assist the OPC UA communication interface.

5 Implementation

In the previous chapter, the approach used at DLR was discussed along with its limitations. This chapter illustrates in detail as to how a mobile roadside traffic setup is made along with its signalling. This is followed by the depiction of a server client setup which makes use of an OPC UA secure connection. The security layer can thus be programmed to the needs of the application to make it more robust while being compliant with international standards.

5.1 Detailed Overview

The approach of the traffic signals first includes the generation of signals. The signals are generated as SPaT messages from the application unit as depicted in the previous chapter. At first the traffic light generates the signals which is sent to Scalance. Scalance is a very secure networking system which uses buses and ports to transfer data from the traffic signal to the Application Unit. The application unit then collects the XML data and encapsulates it into CAM, DENM and SPaT messages. In this project, the SPaT messages are used in which the actual traffic signal information is contained. The OPC server is installed in the Applicaiton Unit which takes over to output the messages from the server to the client. The client machine in this project represents the light signals as the output.

There are a few factors which determine the output of the traffic signals. They are as follows,

- currentState Bitstring of the signal
- IP Address of the server
- Port number for the connection access
- movementState which shows if the signals are interchanging or constant
- the laneNumber for message transfer
- minTimeToChange to depict the minimum time set before the light can change its state.
- maxTimeToChange to depict the longest duration within which the state remains the same before changing to the net state

- likelyTimeToChange to show the actual real time taken for the signal to change considering real world factors such as latency and structural implementations and noise.

There are five main states of the currentState Bitstring signal defined within the mobile traffic signals. The states can either be set or left to alternate in cycles for the duration in which the signal is switched on. There are represented as follows,

1. 1000 : Green
2. 0100 : Yellow
3. 0010 : Red
4. 0110 : Orange
5. 0101/0111 : Flashing

Additionally, the server also uses a port number to communicate the data and relay it to the other components attached. These port numbers act as the bridge connections between the server and connected applications. This thesis uses the port number 8502 for the connection to the local host.

The movement state defines each time the signal is in a particular state. The entirety of the state includes information on the current state and also the lanes. The lane number corresponds to each current state. Each current state has been assigned two lane numbers that are enclosed withint he lane set. The movement state also describes the overall details such as minimum and maximum time to change, real time changes and also the end of the current state and before the start of the next state. Once a movement state is closed it reopens again to describe the next state.

The minimum time to change shows the smallest time before which a change in the colour of the signal is reached. The current frameowrk at DLR used a minimumchange time of eight seconds by default. However, this is not fixed and can be changed to anyduration of time. There is also an option for the user to switch between the states at random times increasing the overall usability of the traffic signal infrastructure.

The maximum time to change depicts the maximum time frame after which the current state of the mobile traffic signal must change. At the moment this value is kept equal to the minimum time to change and can be programmed to any duration of length if and when required. The final part of the output of the traffic signals define the actual time taken for the infrastructure to change from the current state tot he next state.

The next section of the implementation focusses on the OPC UA configuration of the server client with the expectation of finding an alternate to the existing infrastructure tot he one already present at DLR, as discussed in the previous chapter.

The client machine was based on windows and hence an application PuTTY was used as an emulator. The server runs on a linux machine and hence a terminal emulator is the best option to view and relay the results. PuTTY offers several connection means such as telnet, proxy connection or Secure Shell (SSH) connections among others. This thesis uses SSH as the conneciton technique which can be set by the connection type option in the interface. The implementation of PuTTY to connect tot he server is as shown in figure [5.1] below.

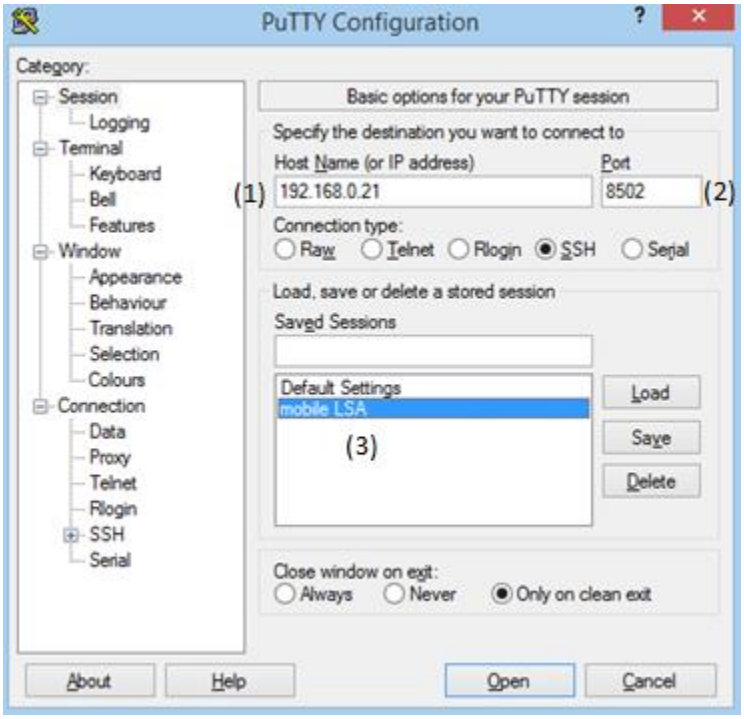

Figure 5.1: Interface of PuTTY

The three important parts involved in the connection are as follows.

1. Host Name/IP address - this depicts the IP address required to connect to the server. In this case, the said server runs on the following IP address and that is :

“92.168.0.21“

There is an option to connect to the server either by using the wireless network or through the ethernet cable. The implemenetation of this thesis uses the wireless network to establish a connection between the server and the client.

2. Port Number : the mobile traffic signal uses ports to specify particular actions performed within the connections. The mobile LSA used in this project used three main ports. They are 8500, 8501 and 8502. Since 8500 and 8501 were pre occupied for other signalling systems, port number 8502 was selected and assigned for connection between the systems.

3. Saved Sessions – the session was then saved once the connection was established. This allowed for an instant connection when ever needed. Also, the ports were now preset to the thesis and blocked for utilization for other projects.

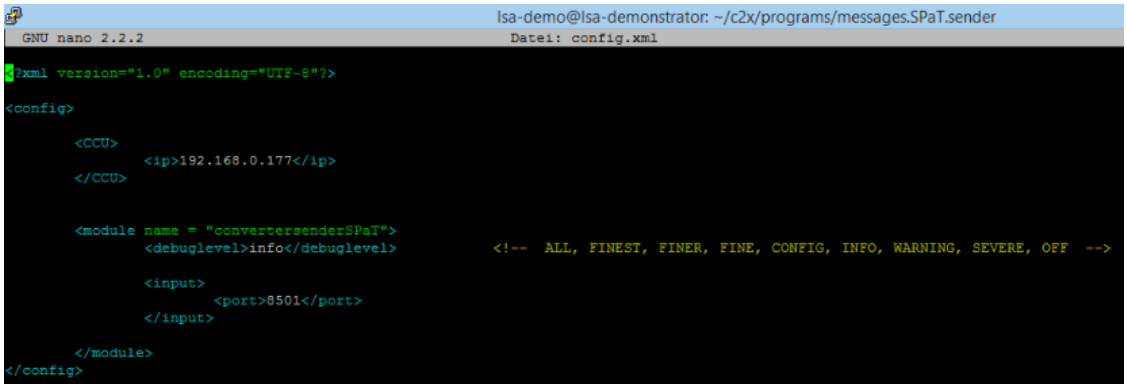

Figure 5.2: Sample configuration window for a port

The OPC Server is then setup on the machine to implement the client-server architecture. The server setup is implemeneted by using JAVA as the programming language. The sequence of execution steps is as follows.

1. The first step is to create a new OPC UA server. The server can then be given a name and used as a point of information transfer.

server = new UaServer();

2. The second step is to describe a networking topology. This thesis uses the IPv4 topology as a standard network topology but an IPv6 topology can be used to define internetworking.

```
server.setEnableIPv6(true);
```

3. The server is then initialized and given a sample name inorder to communicate upon startup. The initialization consists of defining the port numbers and the http port numbers along with the name of he application needed.

```
SampleConsoleServer sampleConsoleServer = new SampleConsoleServer();
sampleConsoleServer.initialize(8502, 52443, APP_NAME);
```

4. Once the initialization is complete, the address space is created. The address space is used by the OPC UA as the means to send and receive data. It consists of three main parts namely object, node and variable.

```
sampleConsoleServer.createAddressSpace();
```

5. The server must now be able to register the clients inorder to form established connections. This is achieved by the means of accepting connection requests with the help of certificates. OPC UA allows its users to register the client certificate and grant trust access manually or it can be done by registering with a universal manager as well. Once the connections are made the the PKI files are used which , in the future validate all trusted clients automatically by skipping the initial setup.

```
final PkiFileBasedCertificateValidator validator = new
PkiFileBasedCertificateValidator();
server.setCertificateValidator(validator);
```

6. The session timer is another aspect which needs to be looked into. This helps kill all ill behaving clients reducing the load on the server.

```
server.getSessionManager().setMaxSessionCount(500);
server.getSessionManager().setMaxSessionTimeout(3600000);
```

The following part of the implementation describs the setup for an OPC UA client. The client can be used for a multiple applications involving data transfer, event subscription, among others. The following steps show the implementation of a OPC UA client and the connection technique used to connect to a server.

1. The first step is to create a new OPC UA client which follows the same method as the server. The client can then be given a name and used as a point of information transfer. We also add a listener to help the client actively lookout for connection acceptances from servers.

```
client = new UaClient(serverUri);
client.setListener(clientListener);
```

2. The client is now initialized with a sample name and assigned an app name which tells the server the appropriate client to connect to. The APP_NAME is the same as the one set in the server initialization.

```
SampleConsoleClient sampleConsoleClient = new SampleConsoleClient();
protected static String APP_NAME = "SampleConsoleClient";
```

3. The certificate must now be created and trusted upon connection with the server. Just like in the server configuratio, the PKI manager can keep record of the clients along with their trusted certificates. This will help the additional overhead of connection requests in the future. The PKI files to keep track of the trusted and rejected server.

```
final PkiFileBasedCertificateValidator validator = new
PkiFileBasedCertificateValidator();
client.setCertificateValidator(validator);
```

4. The nodes and objects are used as the address space definitions are are given by the following,

```
UaNode node = client.getAddressSpace().getNode(nodeId);
```

5. A session timeout can be set inorder to kill all unwanted sessions which may cling to the server thereby increasing more load on it. The set timeouts are in milliseconds and have a default timeout at two minutes. The session timeout has a default of one hour.

```
client.setTimeout(120000); // Default
client.setSessionTimeout(600000);
```

The Client Server relation of the OPC UA is better and easier handled by another application developed by Unified Automation. The application is called UA Expert. The UA Expert provides an excellent interface for connections between the server and the client and also provides an easy access to the address nodes. The address nodes facilitate the transfer of data and files, as described in chapter 4. The following figure [5.3] shows the initial interface on how to create a project and use the server to make appropriate connections. The discovery of the server can also be set with the help of a Local Discovery Server (LDS) Program or it can be configured manually.

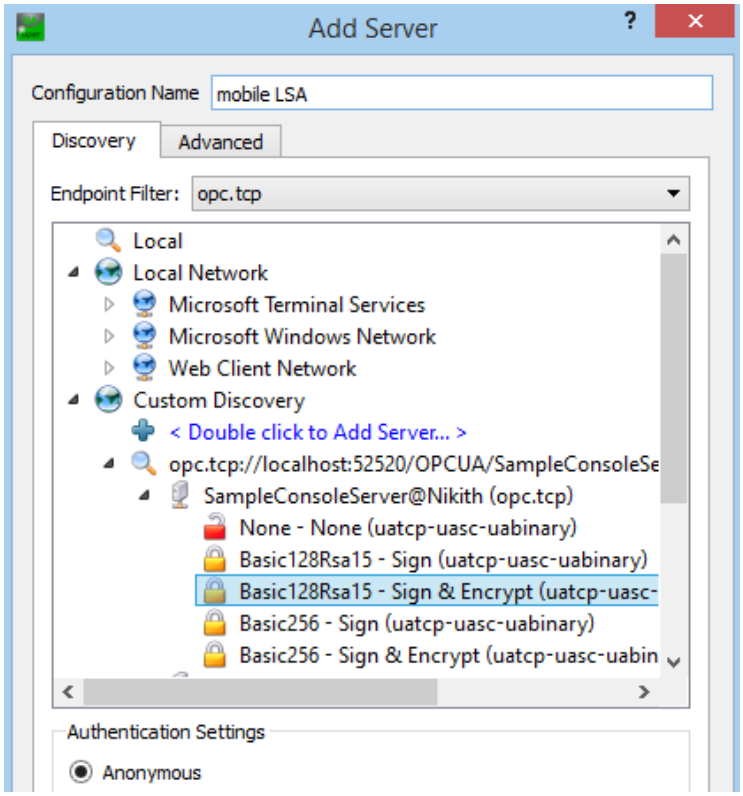

Figure 5.3: Sample configuration window for a port

There are several kinds of encription that can be used inorder to make the connection more secure. This is something that lacks in the existing infrastructure at DLR. There are five kinds of encription that can be chosen. Ther are as follows,

1. No Encryption
2. Basic128Rsa15 – Sign
3. Basic128Rsa15 – Sign and Encrypt
4. Basic 256 – Sign
5. Basic 256 – Sign and Encrypt

These encryption types can also be chosen by the program code within the SDK. However, the availability of the interface makes it more convenient to use. For example, the Basic128Rsa15 is initiated as follows,

```
client.setSecurityMode(SecurityMode.BASIC128RSA15_SIGN_ENCRYPT);
```

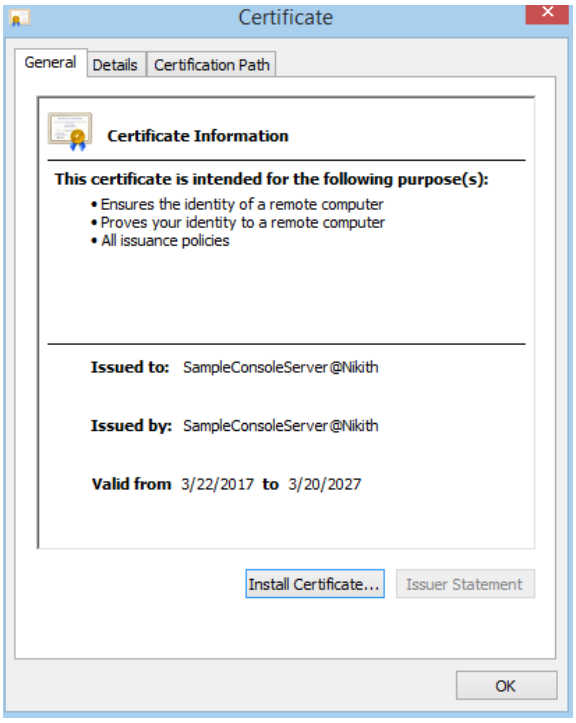

Figure 5.4: Sample of a trusted certificate in the certificate manager

The next step is to accept the certificates and add them to the trusted list so that the client can be connected to the server. The certificate manger keeps a database of all the trusted and non-trusted certificates from all projects. Figure [5.4] is a certificate that is trusted and is encrypted with a Basic128Rsa15 security algorithm.

Once the certificates are taken care of, the connection can be made as shown in Figure [5.5]. After this connection is made, we can then proceed to the Address Space tab in the application. The Address Space enables the movement of both real time data and stored data and also allows for future analysis. Figure [5.6] shows the Address Space pane (lower left window) shows the UA Servers information model. Depending on the Node selected in the Browser the Attribute and Reference Windows (upper right and lower right windows) show the attribute of the selected Node and its references within the meshed network of the servers address space.

You can multi-select UA Nodes in the Address Space window and drag-and-drop them into the DA View. The DA View cerates a subscription and monitors the Nodes. Sampling rate and subscription interval can be changed by right-click into the DA View.

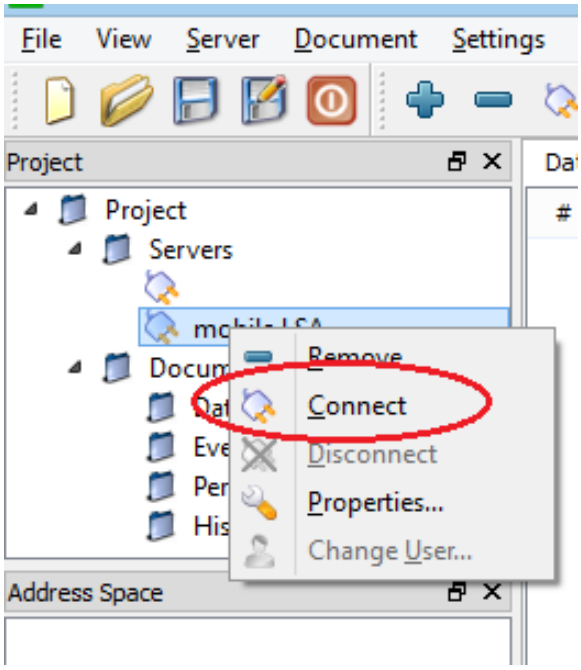

Figure 5.5: Connection between the OPC UA Server and Client

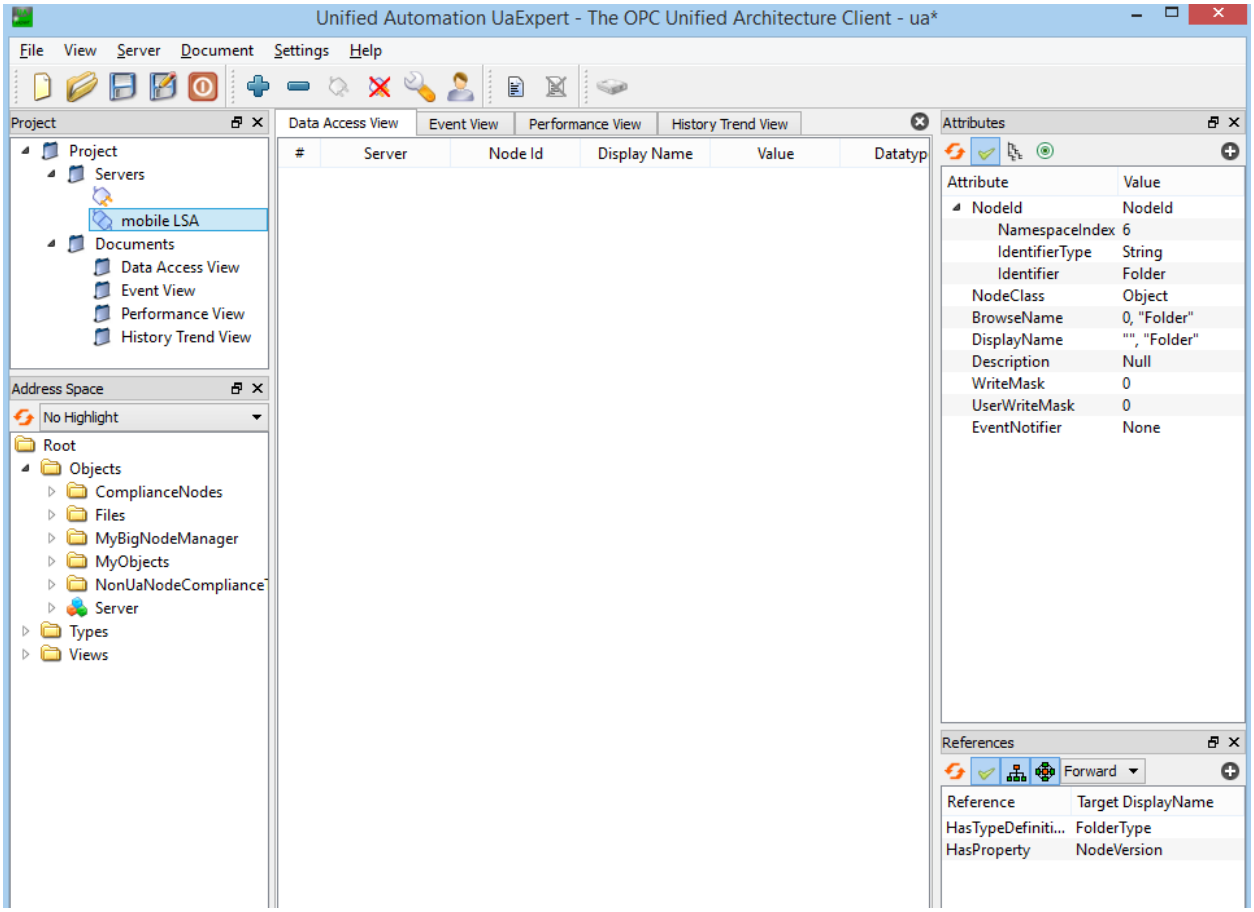

Figure 5.6: Overall view of the UA Expert application

This section has showed the overall implementation of the mobile traffic lights along with the connection to the appropriate devices to get an alternating signal between the current and the next states of the lights. The signal keeps changing until there is an exit condition applied. The later part of the section explains the connetion between the server and the client with the help of a UA Expert application and also witht he help of the server client SDK provided by Unified Automation. The implementation finally shows the usage of a secure encryption channel to send and receive data which makes the data more secure in todays world where data security matters the most, especailly when it pertains to the information exchange in the public domain. OPC UA tends to provide a robust communication medium where the developer can in turn add his own security feature or implement his or her own layer of security for information transfer.

The following section shows the reults of the implementation of all the applications conducted thus far.

5.2 Results

The results are depicted in two parts. The first part is the representation of the output of the Mobile LSA unit which shoes the output of the signal and the traffic light in the real world scenario. The second part shows the implementation of the OPC UA Server-Client architecture where the information is transferred. The results will later be used to infer if this architecture is worth the extra overhead of security to be used in the communication model.

In order to view the results, one must be logged on to the PuTTY. The login status is as shown in Figure 5.7 below:

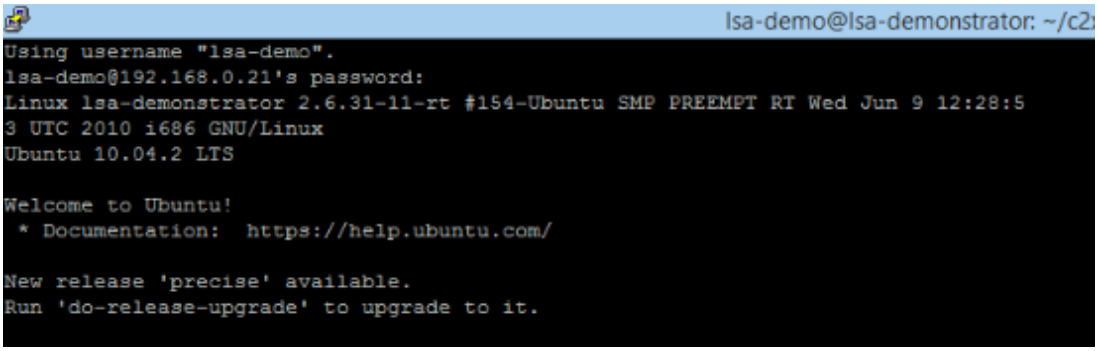

Figure 5.7: Successful login to PuTTY

Once the successful connection is established, the code is run to obtain a signal output in the XML files. The files contain the SPaT messages which depict the output of the traffic lights. The following command in linux renders the signals required.

Sudo tcpdump –I lo –s0 –A udp port 8502

Figure 5.8 shows the output of the above command used. The output is a continuous signal from the application unit. The output signal keeps alternating between its states of colors and the figure shows once such instance. The signal can be switched by using the TCP IP address as well. For example the figure shows a current state of ‘1000’ this means the output of the mobile LSA was ‘Green’ until the next state was received which is 8 seconds later. The next state was ‘0010’ which represents yellow. This state lasted for 9 seconds before the next state was received.

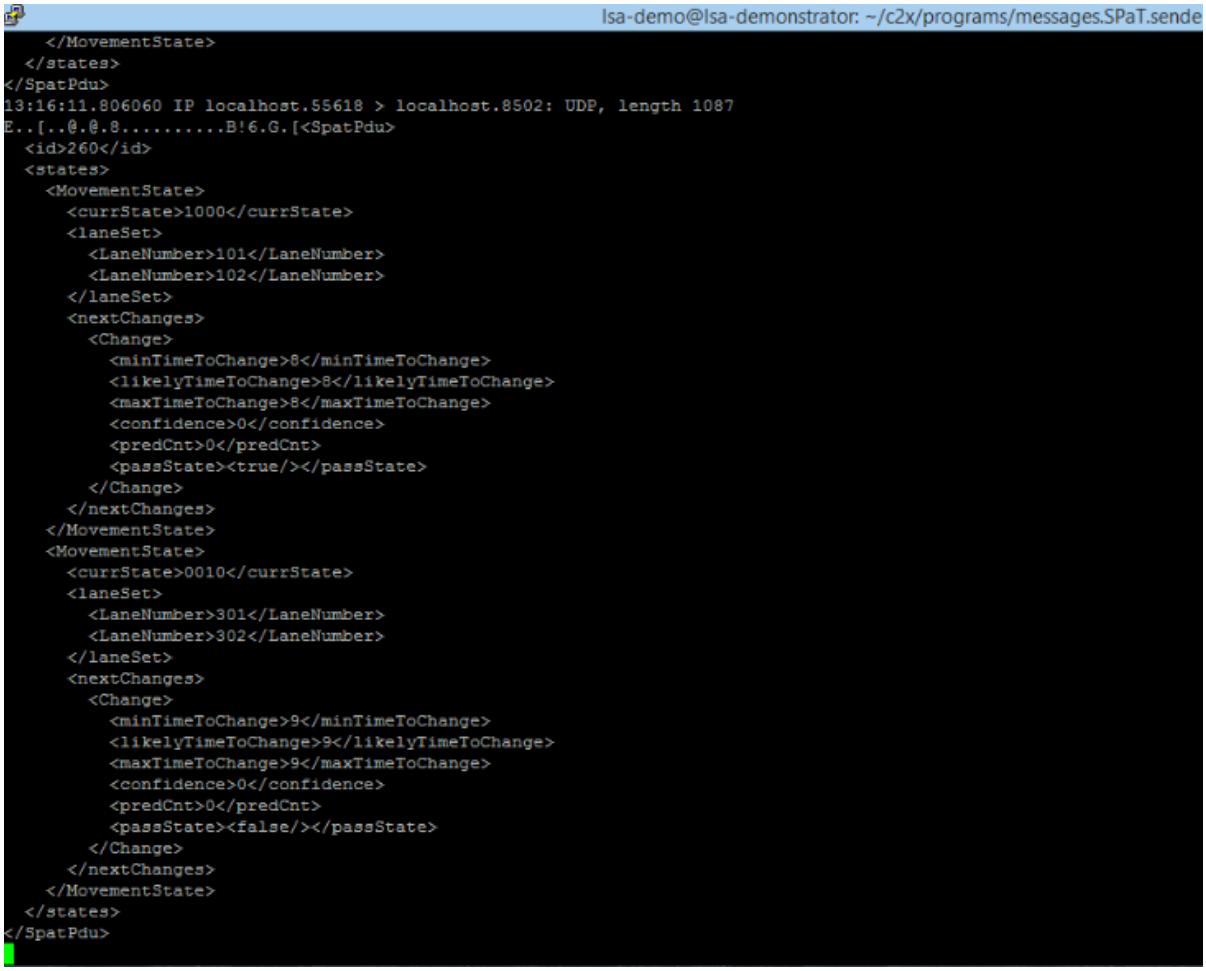

Figure 5.8: Output for the mobile LSA

The next part of the result is from the OPC UA server client configuration from the UA Expert application. To be able to use the application, the connection between the server and the client has to be made at first with the help of the trusted certificates used which are shown in the overview. The certificates are managed universally or they can also be managed manually. Once this connection has been setup, the server cant hen proceed to write and receive data. In the UA Expert, the connection is shown by the clipped result. It can then be used to transfer variables between client and server as shown in Figure 5.10. Figure 5.12 shows the Output Log for the Data Access View. This is just one of the windows and it can be extended to performance views also. Figure 5.13 shows the event log for connection.

This implies that OPC-UA can indeed be used for a secure communication between the server and the client.

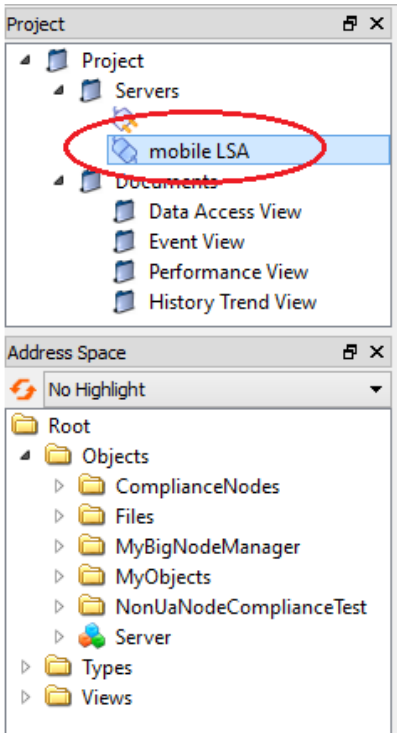

Figure 5.9: Successful connection of the Server Client

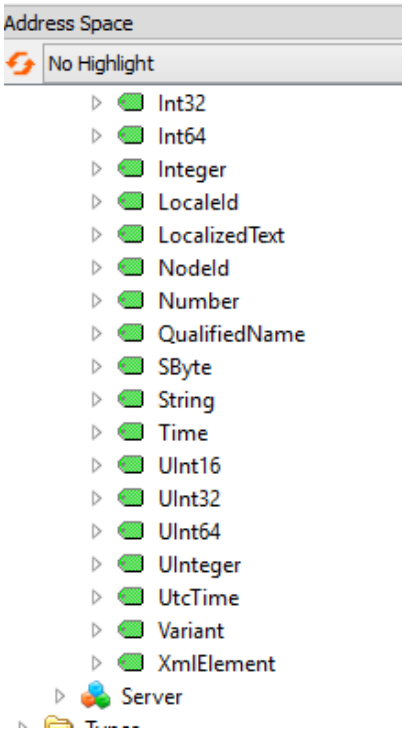

Figure 5.10: Sample of address space types

Data Access View | Event View

#	Server	Node Id	Display Name	Value	Datatype	Source Timestamp	Server Timestamp	Statuscode
1		NS5|String|Data...	DataItem_0000	-95.1056516295	Double	12:59:34.941 PM	12:59:35.360 PM	Good
2		NS2|String|MyL...	MyLevel	11	Double	12:59:34.941 PM	12:59:35.360 PM	Good
3		NS2|String|MyE...	MyEnumObject	1 (One)	Int32	12:49:21.462 PM	12:49:33.360 PM	Good
4		NS0|Numeric|2...	NamespaceArray	{'http://opcfou...	String	12:49:21.392 PM	12:49:33.360 PM	Good
5		NS0|Numeric|2...	ServiceLevel	255	Byte	12:49:19.573 PM	12:49:33.361 PM	Good
6		NS5|String|Data...	DataItem_0000	-95.1056516295	Double	12:59:34.941 PM	12:59:35.360 PM	Good
7		NS0|Numeric|9...	InputArguments	Double click to ...	ExtensionObject	1:00:00.000 AM	12:49:33.361 PM	Good
8		NS5|String|Data...	DataItem_0000	-95.1056516295	Double	12:59:34.941 PM	12:59:35.360 PM	Good
9		NS2|String|MyL...	MyLevel	11	Double	12:59:34.941 PM	12:59:35.360 PM	Good
10		NS2|String|MyL...	MyLevel	11	Double	12:59:34.941 PM	12:59:35.360 PM	Good

Figure 5.11: Successful connection of the Server Client

Log

Timestamp	Source	Server	Message
11/20/2017 12:5...	TypeCache		Reading type info of NodeId NS0|Numeric|35 succeeded
11/20/2017 12:5...	TypeCache		Reading type info of NodeId NS0|Numeric|40 succeeded
11/20/2017 12:5...	Attribute Plugin		Read attributes of node 'NS0|Numeric|85' succeeded [ret = Good].
11/20/2017 12:5...	Reference Plugin		Browse succeeded.
11/20/2017 12:5...	DA Plugin		Item 'NS0|Numeric|9030' succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		DeleteMonitoredItems succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		Item 'NS2|String|MyLevel.Alarm/0:ConfirmedState' succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		DeleteMonitoredItems succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		Item 'NS0|Numeric|2050' succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		DeleteMonitoredItems succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		Item 'NS0|Numeric|3190' succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		DeleteMonitoredItems succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		Item 'NS0|Numeric|2042' succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		DeleteMonitoredItems succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		Item 'NS0|Numeric|2308' succeeded [ret = Good]
11/20/2017 12:5...	DA Plugin		DeleteMonitoredItems succeeded [ret = Good]
11/20/2017 12:4...	AddressSpaceM...		Browse succeeded.
11/20/2017 12:4...	AddressSpaceM...		Browse succeeded.

Figure 5.12: Output Log for the Data Access View

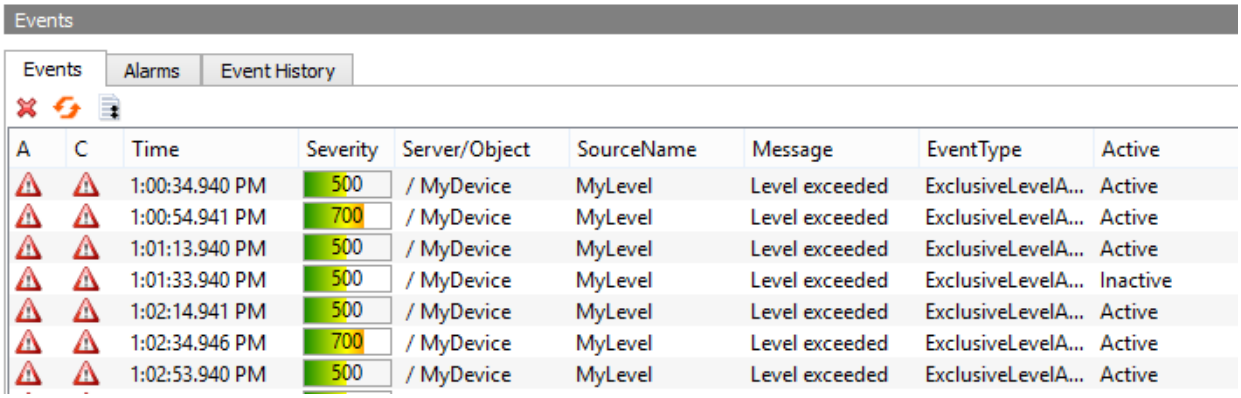

Figure 5.13: Event Log for the connection

6 Summary and Future Work

6.1 Summary

In this chapter we have seen the implementations of OPC-UA. OPC-UA can be used for a secure communication as proposed earlier. This shows that the initial question to the thesis does most likely have a positive result. Upon further research and implementation the OPC-UA will bear its true fruits and help in communication. The usage of OPC UA is a boon for the communication techniques used. A concept proposal is also laid out here.

The concept proposal takes into account the current techniques in detail. The architecture is depicted and the areas of improvement are picked upon for development. The OPC UA in summary has very good advantages not only when single connection requests are made to the server, but also when multiple connections have to be made. This interoperability between machines that use different languages can be a great advantage for any communication model that lays heavy emphasis on a secure transfer of data.

6.2 Future Work

The Future work includes building of seperate layers of security for the OPC-UA. When this is implemented by all the automobile manufacturers, communication between them becomes way more easier. Also, the future of OPC-UA depends on how the industry moves towards. Depending on this direction, the future of this robust, extensible architecture can shape itself.

Bibliography

[1] Hung-An Kao Jay Lee Behrad Bagheri. “A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems”. In: Science Direct (2014).

[2] RTC Magazine. Factory Automation in the World of IIoT. url: http://www. rtcmagazine.com/articles/view/115293.

[3] Boston Consulting Group. Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries. url: https : / / www . bcgperspectives . com / content / articles / engineered _ products _ project _ business _ industry _ 40_future_productivity_growth_manufacturing_industries/.

[4] Bosch Press. Bosch combines “Industrie 4.0” platform and Industrial Internet Consortium standards for the first time. url: http://www.bosch-presse.de/ presseforum/details.htm?txtID=7520&locale=en.

[5] Frank Burkitt. “A strategist’s guide to the Internet of Things”. In: (2015).

[6] Acatech. Recommendations for implementing the strategic initiative INDUS- TRIE 4.0. url: http://www.acatech.de/fileadmin/user_upload/Baumstruktur_ nach_Website/Acatech/root/de/Material_fuer_Sonderseiten/Industrie_ 4.0/Final_report Industrie_4.0_accessible.pdf.

[CAR07] CAR 2 CAR Communication Consortium (C2C-CC): Manifesto. http://www.car-to-car.org/. Version: 2007, Abruf: 21.11.2014

[CAR11] CAR 2 CAR Communication Consortium (C2C-CC): Memorandum of Understanding. http://www.car-to-car.org/. Version: 2011, Abruf: 21.11.2014

[CEN14] CEN/ISO: CEN/ISO TS 19091. Intelligent Transport Systems - Cooperative ITS - Using V2I and I2V Communications for Applications Related to Signalized Intersections. 2014. { Under development

[DLR13] DLR: Anwendungsplattform Intelligente Mobilitat - AIM. http://www.dlr.de/fs/Portaldata/16/Resources/aim/DLR-TS_AnwendungsplattformIntelligenteMobilitaet_2013b.pdf. Version: 2013, Abruf: 11.12.2014

[DoD08] DoD US Department of Defense: Global Positioning System - Standard Positioning Service - Performance Standard. http://www.gps.gov/technical/ps/2008-SPS-performance-standard.pdf. Version: 2008, Abruf: 06.12.2014

[ETS09] ETSI: ETSI TR 102 638. Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; De_nitions. 2009

[ETS10a] ETSI: ETSI EN 302 665. Intelligent Transport Systems (ITS); Communications Architecture. 2010

[ETS10b] ETSI: ETSI TS 102 637-1. Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 1: Functional Requirements. 2010

[ETS10c] ETSI: ETSI TS 102 637-3. Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service. 2010

[ETS11a] ETSI: ETSI TS 102 636-4-1. Intelligent Transport Systems (ITS); Vehicular communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality. 2011

[ETS11b] ETSI: ETSI TS 102 636-5-1. Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 5: Transport Protocols; Sub-part 1: Basic Transport Protocol. 2011

[ETS11c] ETSI: ETSI TS 102 637-2. Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Speci-

_cation of Cooperative Awareness Basic Service. 2011

[ETS12] ETSI: ETSI TS 102 724. Intelligent Transport Systems (ITS); Harmonized
Channel Speci_cations for Intelligent Transport Systems operating in the 5 GHz frequency band. 2012

[ETS13] ETSI: ETSI EN 302 663. Intelligent Transport Systems (ITS); Access layer speci_cation for Intelligent Transport Systems operating in the 5 GHz frequency band. 2013

[GHH08] Gacnik, J. ; Hager, O. ; Hannibal, M.: A Service-Oriented System Architecture For The Human Centered Design Of Intelligent Transportation Systems. In: European Conference

[GHHK08] Gacnik, J. ; Hager, O. ; Hannibal, M. ; K• oster, F.: Service-Oriented Architecture For Future Driver Assistance Systems. In: FISITA
2008 Automotive World Congress, 2008

[IEE12] IEEE Standards Association: IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007). http://standards.ieee.org/getieee802/ download/802.11-2012.pdf. Version: 2012, Abruf: 27.11.2014

[7] L. Atzori, A. Iera, G. Morabito, "The internet of things: A survey", Comput. Netw., vol. 54, no. 15, pp. 2787-2805, 2010.

[8] https://www.car-2-car.org/index.php?id=5

[9] Hagen Stübing "Multilayered Security and Privacy Protection in Car-to-X Networks", 2013

[10] https://opcfoundation.org/

[11] https://www.ibm.com/support/knowledgecenter/en/SSKTWP_8.5.3/com.ibm.notes85.client.doc/sh_object_link_embed_defined_c.html

[12] Tobias Frankiewicz, Arno Hinsberger et. al. Standortbestimmung und Integration von ITS Roadside Stations f¨ur die Anwendungsplattform

Intelligente Mobilit¨at. In AAET: Automatisierungssysteme, Assistenzsysteme und eingebettete Systeme f¨ur Transportmittel, volume 12, pages 27–41. ITS Niedersachsen, Intelligente Transportsysteme und dienste Niedersachsen e.V., 2011.

[13] German Aerospace Center, Institute of Transportation Systems. Application Platform forIntelligent Mobility. www.dlr.de/ts/aim, January 2012.

[14] Tobias Frankiewicz, Lars Schnieder, Frank K¨oster "Application platform for Intelligent Mobility - Test site architecture and Vehicle2X communication setup"

[15] Frankiewicz, T., Schnieder, L., Köster, F., 2012. Application Platform Intelligent Mobility – Test Site Architecture and Vehicle2X Communication Setup, in ITS World Congress, Vienna, Austria.

[16] https://www.unified-automation.com/products/development-tools/uaexpert.html

[17] Burmeister, Alexander (2015) Entwicklung und Realisierung eines Messverfahrens zur Untersuchung und Bewertung der Dienstgüte-Eigenschaften eines C2X-Kommunikationssystems.

[18] Mikat, Jürgen and Brockfeld, Elmar and Wagner, Peter (2003) Agent Based Traffic Signals Regulating Flow On a Basic Grid. Agent-Based Simulation 4, 2003-04-28

[19] Mahnke, Wolfgang, Leitner, Stefan-Helmut, Damm, Matthias “OPC Unified Architecture”