Bachelorarbeit

Frau
Xiaoning Zhou

Konzeption und Umsetzung eines Komplexpraktikums zur Car-2-X-Netzwerksimulation unter Verwendung des Simulationswerkzeugs CANoe

2021

Fakultät: Ingenieurwissenschaften

BACHELORARBEIT

Konzeption und Umsetzung eines Komplexpraktikums zur Car-2-X-Netzwerksimulation unter Verwendung des Simulationswerkzeugs CANoe

Autorin:
Frau Xiaoming Zhou

Studiengang:
Elektro- u. Informationstechnik

Seminargruppe:
EI17sA-BC

Erstprüfer:
Prof. Dr.-Ing. Jan Thomanek

Zweitprüfer:
M. Sc. Hongwei Xu

Einreichung:
Mittweida, 08.03.2021

Faculty of Engineering

BACHELOR THESIS

Conception and implementation of a complex practical course on Car-2-X network simulation using the simulation tool CANoe

author:
Ms. Xiaoming Zhou

course of studies:
Electrical and Information Technology

seminar group:
EI17sA-BC

first examiner:
Prof. Dr.-Ing. Jan Thomanek

second examiner:
M. Sc. Hongwei Xu

submission:
Mittweida, 08.03.2021

Bibliografische Angaben

Xiaoming, Zhou:

Konzeption und Umsetzung eines Komplexpraktikums zur Car-2-X-Netzwerksimulation unter Verwendung des Simulationswerkzeugs CANoe

Conception and implementation of a complex practical course on Car-2-X network simulation using the simulation tool CANoe

70 Seiten, Hochschule Mittweida, University of Applied Sciences,
Fakultät Ingenieurwissenschaften, Bachelorarbeit, 2021

Referat:

Die vorliegende Arbeit befasst sich mit der Erstellung von Praktikumsversuchen im Rahmen der studentischen Ausbildung zur Thematik der Fahrzeugvernetzung (Car-2-X).

Inhaltsverzeichnis

Abkürzungsverzeichnis

ADAS	Advanced Driver Assistance Systems
BSM	die Basic Safety Message
CAM	Cooperative Awareness Message
CAN(FD)	Controller Area Network(Flexible Data Rate)
CAV	vernetzte autonomen Fahrzeuge
CEN	Comité Européen de Standardization
C-ITS	Kooperative-Intelligent Transportation System
DENM	Decentralized Environmental Notification Messages
DSRC	Dedicated Short Range Communication
DUT	Device Under Test
ETSI	European Telecommunications Standards Institute.
IEEE	Institute of Electrical and Electronics Engineers
ISM	Industrial, Scientific and Medical
ITS-AID	ITS Application Identifier
KQA	Kreuzungs-/Querverkehrsassistent
LIN	Local Interconnect Network
LKW	Lastkraftwagen
MAC	Media Access Control
OSI	Open Systems Interconnection
PKW	Personenkraftwagen
PSID	Provider-Dienstes, zugewiesen Identifier

RSU	Road Side Units
SAE	Society of Automotive Engineers
SDOs	Standards Developing Organizations
SPAT	Signal Phase and Timing
TC	Technisches Komitee
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
V2X	Vehicle to Anything
VANETs	Vehicular ad hoc networks
WAVE	Wireless Access in Vehicular Environment
WGs	Arbeitsgruppen
WSMP	WAVE Short Message Protocol

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Im einleitenden Kapitel werden die Motivation und die Aufgabenstellung dieser Bachelorarbeit besprochen. Gleichzeitig erfolgt ein kurzer Überblick zu den einzelnen Kapiteln dieser Arbeit.

1.1 Motivation

Mit dem Begriff autonomes Fahren beschreibt man in der Regel selbstfahrende Fahrzeuge oder Transportsysteme, die sich ohne Eingriff des menschlichen Fahrers zielgerichtet fortbewegen. Wie die Entwicklungsstufen zu einem solchen vollautonomen Fahrzeug aussehen, hat SAE International (Society of Automotive Engineers) 2014 mit dem J3016-Standard definiert. Die Stufen des autonomen Fahrens reichen von Stufe 0 (keine Automatisierung) bis hin zu Stufe 5 (fahrerloses Fahren). (vgl. Abbildung 1).

Stufe	Stufe 0	Stufe 1	Stufe 2	Stufe 3	Stufe 4		Stufe 5	
Bezeichnung (BASt/ VDA) SAE-Bezeichnung	nicht automatisiert no automation	assistiert driver assistance	teil-automatisiert partial driving automation	hoch-automatisiert conditional driving automation	voll-automatisiert high driving automation		domänen-unabhängig full driving automation	
Ausführung der Fahraufgabe	Fahrer	**System** führt Längs- **oder** Querführung aus	**System** führt Längs- **und** Querführung aus	**System** führt Längs- **und** Querführung aus	**System** führt Längs- **und** Querführung aus	**System** führt Längs- **und** Querführung aus	**System** führt Längs- **und** Querführung aus	**System** führt Längs- **und** Querführung aus
Überwachung der Fahraufgabe	dauerhaft durch **Fahrer**	dauerhaft durch Fahrer	dauerhaft durch Fahrer	während der automatisierten Fahrt durch **System**	während der automatisierten Fahrt durch **System**	dauerhaft durch das **System**	während der automatisierten Fahrt durch **System**	dauerhaft durch das **System**
Rückfallebene im Fehlerfall	Fahrer	Fahrer	Fahrer	**Fahrer** wird mehrmals zur Übernahme aufgefordert	System	System	System	System
Anforderungen an den Fahrer	dauerhafte **Ausführung** der Fahraufgabe	**dauerhafte Überwachung** und **Ausführung** der Quer- **oder** Längsführung	**dauerhafte Überwachung** und jederzeit **Bereitschaft** zur vollständigen Übernahme	**Bereitschaft**, die Fahraufgabe nach Aufforderung zu übernehmen	**keine**, Fahrer kann auf Wunsch das System abschalten und selbst fahren	**kein Fahrer im Fahrzeug**, ggf. Lotse außerhalb des Fahrzeugs	**keine**, Fahrer kann auf Wunsch das System abschalten und selbst fahren	**kein Fahrer im Fahrzeug**, ggf. Lotse außerhalb des Fahrzeugs
Beispiel	Fahrer fährt	Abstandsregel-tempomat	Stauassistent unterstützt Längs- und Querführung auf Autobahnen	Autobahn-System übernimmt zeitweise Längs- und Querführung auf Autobahnen	Autobahn-System übernimmt Längs- und Querführung auf Autobahnen	fahrerloses Shuttle auf definierten Strecken in Stadt XY	System übernimmt Längs- und Querführung überall	fahrerloses Shuttle, das überall fahren kann

Abbildung 1: SAE-Stufen zum Grad des automatisierten Fahrens [1]

Beispiele von aktuellen Anwendungen und Assistenzfunktionen sind in Abbildung 2 dargestellt.

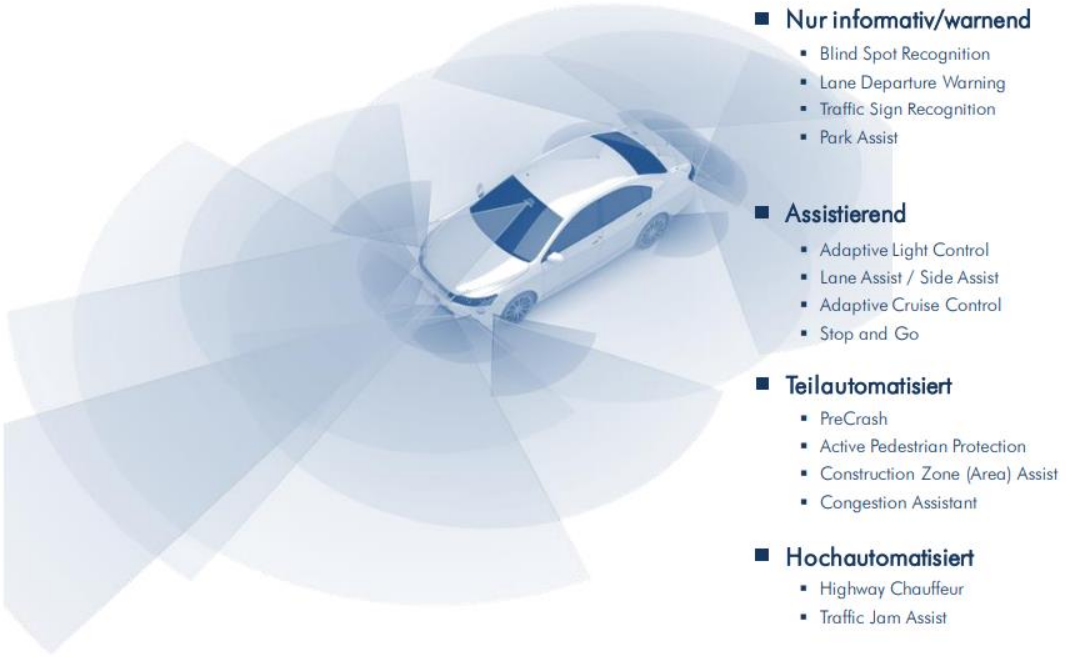


Abbildung 2: Assistenzfunktionen klassifiziert nach dem Automatisierungsgrad [Quelle: IAV GmbH]

Moderne Fahrzeuge werden zunehmend mit sensorischen Komponenten ausgestattet. Durch die Anwendung von Advanced Driver Assistance Systems (ADAS), bis hin zum voll-autonomen Fahren, werden in Zukunft aus PKWs und LKWs fahrende Messstationen.

Vehicle-2-X ist als Erweiterung für die Sensoren des Fahrzeugs. Ein Sensor kann dabei auch eine V2X-Einheit im Fahrzeug sein, die Daten von anderen Fahrzeugen oder sogenannter Road Side Units (RSU) empfängt und damit den eigenen Erfassungshorizont erweitert.[2]

Für eine vollständige Erfassung der umgebenden Szene stoßen Einzelsensorlösungen jedoch schnell an ihre Grenzen. Das Fahrzeug erhält damit Informationen über seine Umgebung, die die Sensoren nicht erkennen können. Zu Beispiel: Es gibt Unfall auf der Straße vor uns. Eine automatische Notbremsung können gegebenenfalls im Assistenzsysteme eingeleitet werden. Assistenzsysteme sind, die auf die Daten von mehr als einem Sensor zurückgreifen können, besser geeignet, um das Unfallfahrzeug zu erkennen.

1.2 Zielsetzung

In dieser Bachelorarbeit wird Konzeption und Umsetzung eines Komplexpraktikums für die Studentische Ausbildung zur Car-2-X-Netzwerksimulation unter Verwendung des Simulationswerkzeugs CANoe vorgestellt, weil diese Manöver ohne Automatisierung zur steigenden Zahl von Unfällen führen. Beispiele für die Anwendungen der V2X sind: automatische Geschwindigkeitsanpassung an einen Führer auf der Autobahn folgen, automatische Einfahrt auf der Autobahn oder Selbstparken. Mit der Anwendung der V2X werden die Verkehrsüberlastung und Umweltverschmutzung reduzieren. V2X kann also anderer Informationsdienste bereitstellen. Und das wichtigste Ziel der Arbeit ist, toolgestützte Praktikumsversuche für V2X-Anwendungen zu entwickeln. Es werden mit CANoe Software, VN4610 und CAPL-Programmierung den Straßenzustand, die Bewegungen und Zustand der Fahrzeuge und Fahrzeugkommunikation simulieren, realisieren und testen. Um solche Aufgaben zu erfüllen, wäre ein großer Schritt zu tun, die Umweltwahrnehmung der Umgebung durch Fahrzeugkommunikation in dem sogenannten kooperativen Ansatz.

1.3 Kapitelübersicht

Zunächst werden in Kapitel 2 die Grundlagen zu V2X näher erläutert und die Anwendung der CANoe Software wird erklärt. Danach wird VN4610 Netzwerk Interface und CAPL Programmierung erwähnt. In Kapitel 3 wird die Erstellungen von Praktikumsversuchen dargestellt. In Kapitel 4 werden die Versuche evaluiert. In Kapitel 5 werde die Arbeit mit einem Fazit enden. Außerdem wird einen Ausblick gemacht.

2 Grundlagen zu V2X

Das Kapitel startet mit dem Beispiel zu V2X. Dann werden die entsprechenden Standards in EU, USA, CN vorgestellt. Danach werden Physical Layer und den Aufbau der Botschaften erwähnt. Außerdem wird CANoe Software vorgestellt. Schließlich wird VN4610 Netzwerk Interface und CAPL Programmierung erläutert.

2.1 Anwendungsfälle zu V2X

Die Kommunikation zwischen zwei Fahrzeugen wird „Vehicle to Vehicle-Communication" (Vehicle2Vehicle oder kurz V2V) genannt. Die Kommunikation zwischen einem Fahrzeug und der Infrastruktur wird „Vehicle to Infrastructure-Communication" (Vehicle2Infrasturcture oder kurz. V2I) genannt. Allgemein wird die Kommunikation von Fahrzeugen „Vehicle to Anything-Communication" (Vehicle2Anything kurz V2X) genannt.[3]

2.1.1 Hinderniserkennung

Die Hinderniswarnung warnt z. B. vor liegengebliebenen Fahrzeugen oder Personen, Tieren und Gegenständen auf der Fahrbahn; für letztgenannte Fälle kann die Detektion z. B. einfach durch den Fahrer erfolgen, indem Warnmeldungen bzgl. erkannter Gefahren durch manuelle Interaktion über das HMI ausgelöst werden. Verfügt das Fahrzeug z. B. auch über eine Kamera mit Bildverarbeitung, so ließe sich dieser Vorgang auch automatisieren. Hindernisse auf der Fahrbahn können auch aus der Analyse von Ausweichmanövern detektiert werden. Wird ein Fahrzeug aufgrund einer Panne oder eines Unfalls selbst zum Hindernis, so erfolgt die Detektion automatisch aus der Analyse verschiedener CAN-Daten. Im einfachsten Fall detektiert sich ein liegengebliebenes Fahrzeug dadurch, dass es steht und der Warnblinker aktiv ist, s. dazu. Abb.3. Das Fahrzeug am linken Bildrand ist liegengeblieben und der Warnblinker ist aktiv. Eine detektierte Gefahr führt zum Versand einer DENM, die in nachfolgenden Fahrzeugen empfangen wird: Sie enthält unter anderem Angaben zu Hindernistyp, -Zeitpunkt, -position und Verbreitungsgebiet der Nachricht. Alle Fahrzeuge, die sich innerhalb der Kommunikationsreichweite des Senders befinden, empfangen die Nachricht direkt. Für weiter entfernte Empfänger ist dies nicht garantiert und die Nachricht wird mittels des Multi-Hop-Verfahrens über mehrere Fahrzeuge weitergereicht, was durch die gelbe Linie angedeutet ist. Empfangende Fahrzeuge prüfen, ob sie auf das Hindernis zufahren (räumliche Relevanz) und ob Dringlichkeit (zeitliche Relevanz) gegeben ist, also ob eine Information oder eine Warnung auf dem HMI angemessen ist. Dabei kann die räumliche Relevanz durch Abgleich der Hindernisposition mit der

Fahrzeugroute anhand einer digitalen Karte oder durch Abgleich der aktuellen Empfängerposition mit einer in der DENM enthaltenen Positionskette (Abfolge historischer Positionen des Sendefahrzeugs) geprüft werden, s. Abb. 3 rechts. Der erhaltene Abstand zum Hindernis erlaubt mithilfe der Geschwindigkeit die Bestimmung einer „time-to-obstacle“, anhand derer die zeitliche Relevanz bestimmt wird. [4]

Abbildung 3: Anwendungsbeispiel Hinderniswarnung [Quelle: Bosch]

2.1.2 Kreuzungs-/Querverkehrsassistent

Heutzutage sind die Fahrzeuge mit zahlreichen Sicherheitssystemen und Sensoren ausgestattet, mit Hilfe dieser werden die Umgebung und Informationen über den aktuellen Fahrzustand erfasst. Diese Daten werden vom Fahrzeug zusammengefasst und über V2X verbreitet. Somit können Fahrzeuge andere Verkehrsteilnehmer über mögliche Gefahrenstellen, wie Glatteis auf der Straße, Aquaplaning, Staus, Unfälle oder andere gefährliche Verkehrssituationen warnen, bevor diese mit ihnen konfrontiert werden. Hier ist das Beispiel über die Kreuzung/ Querverkehrsassistent.

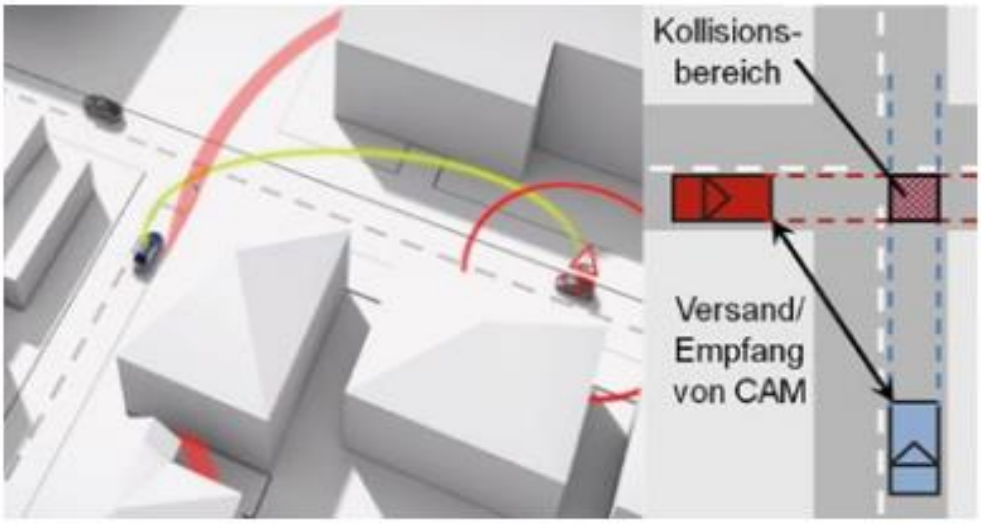


Abbildung 4: 8 Anwendungsbeispiel Kreuzungs-/Querverkehrsassistent [Quelle: Bosch]

Der Kreuzungs-/Querverkehrsassistent (KQA) informiert bzw. warnt den Fahrer im Falle einer möglichen Kollision mit Abbiege- oder Querverkehr an Kreuzungen und Einmündungen. Hierzu senden die Fahrzeuge im Anfahrts- und Innenbereich einer Kreuzung in ausreichend dichtem Zeittakt CAMs mit Positions- und Bewegungsdaten (Geschwindigkeit, Fahrtrichtung, …) und empfangen jene Daten von anderen Fahrzeugen. Die Bewegung des eigenen und der anderen Fahrzeuge werden prädiziert und daraus ein Kollisionsrisiko bestimmt.

Zunächst wird hierzu – wie in Abb. 4 rechts dargestellt – ein Kollisionsbereich je Sendefahrzeug bestimmt. „Dies ist im Wesentlichen der Ort, an dem sich die Fahrtrajektorien der Fahrzeuge kreuzen. Biegt das rote Fahrzeug rechts ab, so gibt es keinen solchen Bereich; ist er jedoch vorhanden, wird geprüft, ob die beteiligten Fahrzeuge sich ungefähr gleichzeitig in ihm befinden werden oder ob eines der beiden Fahrzeuge ihn wesentlich früher als das andere Fahrzeug passiert. In diesem Fall wird im nicht vorfahrtsberechtigten Fahrzeug die Zeit bestimmt, bis es diesen Bereich erreicht; beim Unterschreiten einer kritischen Zeitschwelle wird der Fahrer rechtzeitig gewarnt. Diese Bestimmung geschieht unter Auswertung der Fahrabsicht, z. B. wird bei einer begonnenen Bremsung nicht gewarnt.[4]

2.1.3 „Grüne Welle“

Basierend auf den Daten aus dem Feldversuch – unterstützt durch Fahr- und Verkehrssimulationen – konnten weitere positive Einflüsse auf die Fahr-bzw. Verkehrssicherheit und -effizienz ermittelt werden: So verschob sich bei der Hinderniswarnung der Bremszeitpunkt um bis zu 50 m nach vorne und die Gefahrenstelle wurde um bis zu 15 km/h langsamer passiert; beim elektronischen Bremslicht wurde eine

Verbesserung der Reaktionszeit für nachfolgende Fahrzeuge um 60 % beobachtet. Hinsichtlich einer Steigerung der Verkehrseffizienz ist der „LSA-Phasenassistent“ („Grüne-Welle-Assistenz“) hervorzuheben: In einer mit Feldversuchsdaten kalibrierten Verkehrssimulation konnte gezeigt werden, dass die Verlustzeiten und die Anzahl der Halte abnehmen. Diese Effekte treten schon bei Ausstattungsraten von 5 % auf und nehmen bei höheren Ausstattungsraten noch zu; darüber hinaus sinkt die Häufigkeit von Geschwindigkeitsüberschreitungen in der Kreuzungsanfahrt.[4]

2.2 Standards

Die Standards spielen eine große Rolle in Erleichterung der Annahme von einer Technologie, während Interoperabilität zwischen Produkten der unterschiedlichen Hersteller ermöglicht. Die Standardisierung intelligenter Verkehrssysteme (ITS) beinhaltet Interessenträger in verschiedenen Regionen. Zur Ermöglichung eines globalen interoperablen ITS-Systems, arbeiten die EU, Japan und die USA eng zusammen, um harmonisierte Normen zu erreichen. Ein kompakter Überblick über einen konsistenten Satz wichtiger Standards für den Bau von ITS-Stationen Referate für Fahrzeug ad-hoc-Netze (VANETs) und Kooperative-ITS (C-ITS) und in allgemein für ITS, ist in diesem Kapitel vorgesehen. Zu einem großen Teil werden Verweise gemacht Normen von CEN, ETSI, IEEE, ISO und SAE.[5]

Unter den staatlichen SDOs (Standards Developing Organizations) ist die International Technisches Komitee (TC) 204 "Intelligenter Verkehr Systeme" den Wichtigste Verfechter der ITS-Standards. ISO/TC 204 mit ihren 18 Arbeitsgruppen (WGs) ist in 1992 gegründet, was verantwortlich für die Gesamtsystem- und Infrastrukturaspekte von ITS unter Berücksichtigung der Arbeiten bestehenden internationalen Normungsgremien ist. ISO TC2 04 kooperiert mit dem Comité Européen de Standardization (CEN) TC 278 ITS,3 Entwicklung gemeinsam CEN/ISO-Normen. (Siehe Abb. 5)

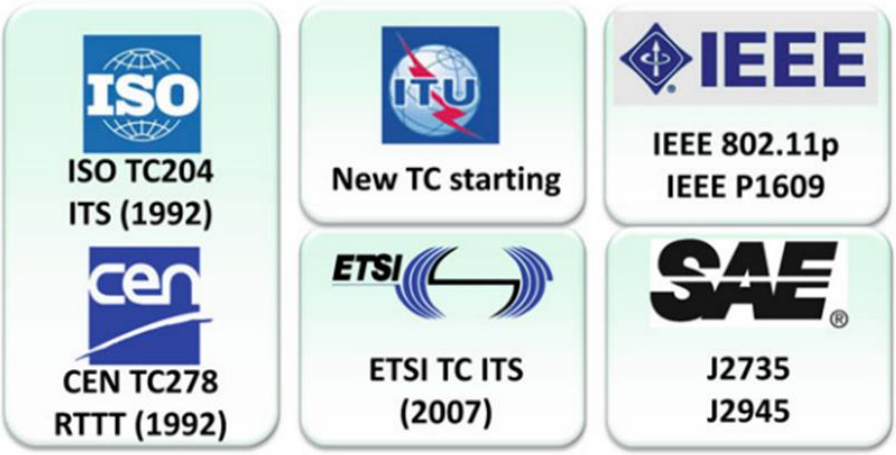


Abbildung 5: Standard Development Organizations developing ITS [5]

Unterschiedliche Staaten verwenden unterschiedlichen Standards.

2.2.1 EU ETSI ITS G5

Das Europäische Institut für Telekommunikationsnormen (ETSI) ist eine Normungsorganisation für Informations- und Kommunikationstechnologien in Europa. Das ETSI standardisiert die V2X-Kommunikation aufbauend auf dem 802.11p Standard in dem Cooperative-Intelligent Transportation System (C-ITS).

Die ITS-Stationsreferenzarchitektur verwendet eine vereinfachte und erweiterte Open Systems Interconnection (OSI)-Modell. Die Kommunikationsschichten sind die ITS-S Zugriffsschicht (OSI-Schichten 1 und 2), die ITS-S Networking & Transport-Schicht (OSI Layer 3 und 4) und die ITS-S-Einrichtungen-Schicht (OSI-Layer 4, 5 und 6). Oben auf dem Kommunikationsebenen ist die ITS-S Applications-Entität. (Siehe Abb. 6)

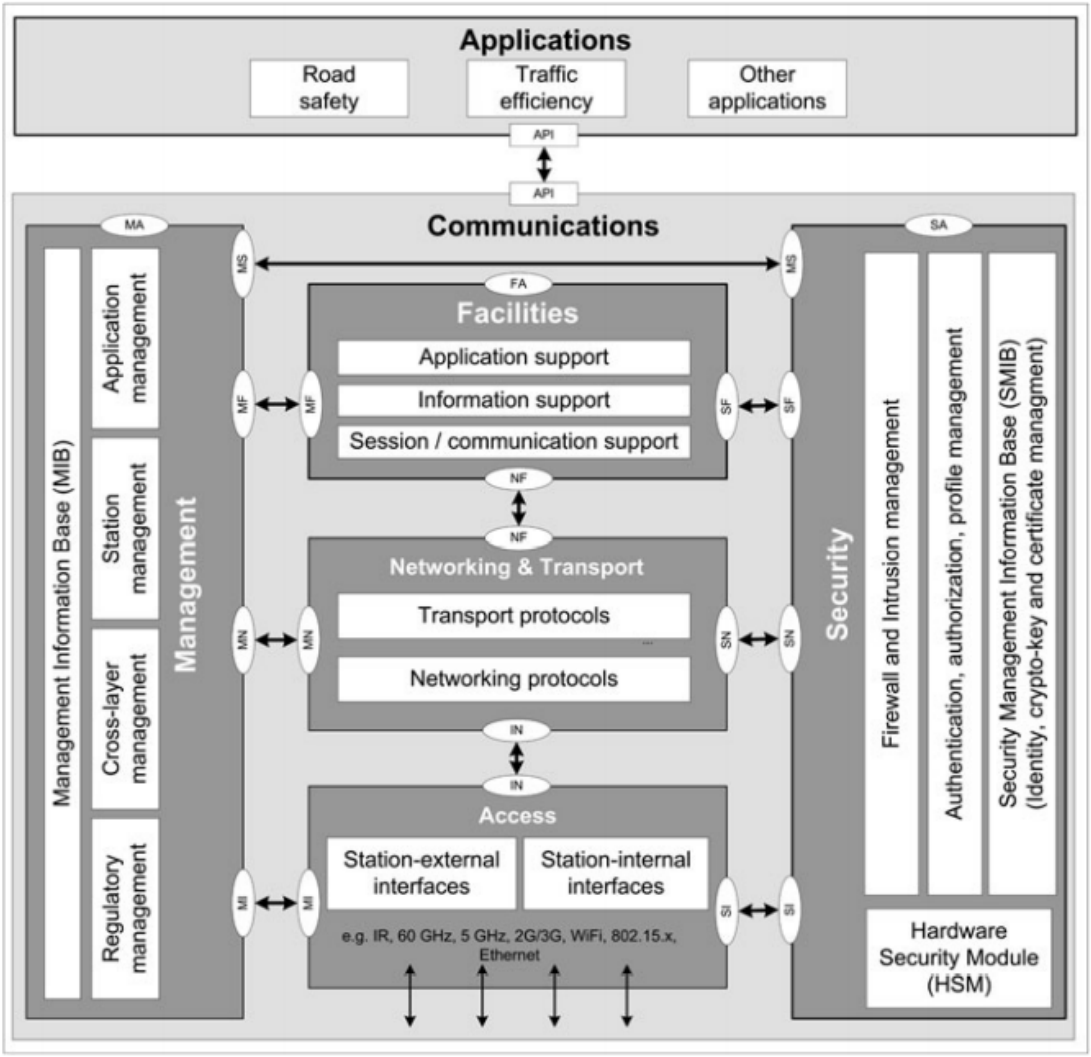


Abbildung 6: Architektur der ITS-Station [6]

2.2.2 USA WAVE/ IEEE1609

Institute of Electrical and Electronics Engineers (IEEE) Standards wie IEEE 802.11 werden für ITS verwendet, und IEEE 1609 WG5 (Securit) entwickelte eine Reihe von Standards unter den Arbeitstitel "Wireless Access in Vehicular Environment" (WAVE), der ein optimiertes Subsystem von ITS mit einer spezifischen eigenen Architektur und einem Schwerpunkt auf IEEE 802.11 Zutrittstechnologie.

Die WAVE-Gerätearchitektur ist in IEEE 1609.0 standardisiert. Diese Architektur von Absicht stellt nur einen Teilsatz der Funktionalität einer ITS-Station bereit. WAVE basiert auf IEEE Std 802.11 und ist für schnelle Zuverlässigkeit optimiert, Z.B bei der Übertragung von Sicherheitsmeldungen. (Siehe Abb.7)

Die 1609 Familie von Standards umfasst:

- IEEE Std. 1609.0, Architektur. Es beschreibt, wie die IEEE 1609-Standards funktionieren zusammen.

- IEEE Std 1609.2, Sicherheitsdienste für Anwendungen und Management-Nachrichten. Es definiert sichere Nachrichtenformate und Verarbeitung.

- IEEE Std 1609.3, Netzwerkdienste. Es definiert Netzwerk- und Transportschicht einschließlich des "WAVE Short Message Protocol" (WSMP) für effiziente Single-Hop-Null-Netzwerk-Kommunikation und gewöhnliche IP-Adressierung und Routing (ohne iso-standardisierte Mobilitätsmerkmale).

- IEEE Std 1609.4, Mehrkanalbetrieb. Es bietet Verbesserungen an der IEEE 802.11 Media Access Control (MAC) zur Unterstützung von Mehrkanal-WAVE Operationen.

- Entwurf von IEEE P1609.6, Remote Verwaltungsdienste. Es bietet interoperable Dienste zum Verwalten von WAVE-Geräten, die IEEE Std 1609.3.

- IEEE Std 1609.11, Over-the-Air Electronic Payment Data Exchange Protocol für ITS. Es definiert die Dienste und sichere Nachrichtenformate zur Unterstützung von secure elektronische Zahlungen.

- IEEE Std 1609.12, Identifier Allocations. Er gibt Bezeichnerwerte an, die WURDEN für die Verwendung durch WAVE-Systeme, einschließlich des Provider-Dienstes, zugewiesen Identifier (PSID) Zuordnungen harmonisiert mit der ITS Application Identifier (ITS-AID) verwendet in ISO, CEN und ETSI, ISO, CEN und ETSI.[5]

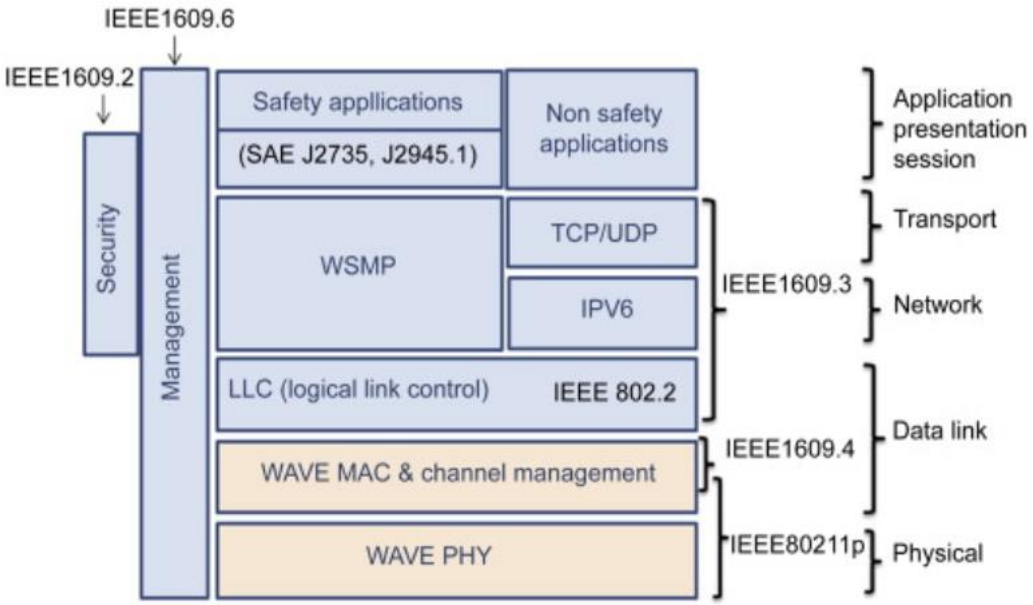


Abbildung 7: Protokoll-Stacks von WAVE [19]

2.2.3 CHINA Standard

Im Juni 2018 veröffentlichte MIIT den Leitfaden für den Bau des nationalen ICV-Standardsystems (Informations- und Kommunikationsabteilung). Dieser Leitfaden ist ein Planungsdokument der Regierung für Chinas Standardsystem für Information und Kommunikation von vernetzten Fahrzeugen und hat großen Einfluss auf die Normung in diesem Bereich. Demnach ist Chinas Standardsystem der vernetzten Fahrzeugindustrie (Information und Kommunikation) hauptsächlich aus vier Teilen, darunter die meisten Grundnormen, Kommunikationsprotokolle und Ausrüstungen, Kommunikationsdienste und Anwendungstechnologien sowie Netzwerk- und Datensicherheitsstandards. Unter ihnen, Kommunikationsprotokolle und Gerätetechnikstandards betreffen hauptsächlich LTE-V2X-Technologie, 5G eV2X Satellitenkommunikations-, Navigations- und Ortungstechnik sowie Fahrzeugkommunikation Gerätetechnik. Dieser Leitfaden enthält keine IEEE 802.11p-Technologie. Im Vergleich dazu kann C-V2X einen höheren Kommunikationsbereich bieten. Der C-V2X ist für die direkte Kommunikation mit niedriger Latenz ausgelegt. Sicherheitsmeldungen, wie z. B. Gefahrenwarnungen im Straßenverkehr, können über eine direkte Kommunikation mit geringer Latenz Übertragung im weltweit konsistenten 5,9 GHz ITS-Spektrum gesendet werden. Der C-V2X hat eine minimale Übertragungsverzögerung von bis zu 4ms und kann geringer sein. Im Gegensatz zu IEEE 802.11p-basierten Technologien müssen C-V2X-Anbieter von Direktkommunikationsregistern diese Spezifikationen erfüllen, um eine vorhersagbare und einheitliche Leistung zu erzielen. Da 802.11p keine Mindestleistungsanforderungen festgelegt, ist seine Leistung möglicherweise nicht vorhersagbar und eignet

sich nicht für Sicherheitsanwendungen in realen Automotive-Bereitstellungsszenarien. 802.11p ist anfällig für Interferenzen, da der fehlende symbolische Querzugriff Übertragungen anfällig für kurze Impulse macht. In Bezug auf die LTE-V2X-Technologie umfasst das Internet der Fahrzeuge Kommunikationsstandards in diesem Leitfaden vorgeschlagen:

LTE-V-Technologiestandards, Schnittstellenstandards, Endgerätestandards, Netzwerkausrüstungsstandards, Standards für Netzwerkschicht/Anwendungsschicht und Interoperabilitätsstandards.[18]

2.3 Schicht 1 – Bitübertragungsschicht (Physical Layer)

Der Physical Layer ist die unterste Schicht im OSI-Modell. Die Bitübertragungsschicht definiert die elektrische, mechanische und funktionale Schnittstelle zum Übertragungsmedium. Die Protokolle dieser Schicht unterscheiden sich nur nach dem eingesetzten Übertragungsmedium und -verfahren. Das Übertragungsmedium ist jedoch kein Bestandteil der Schicht 1.[7]

Um kooperative ITS-Dienste zu ermöglichen, wurden in Europa 30 MHz Bandbreite unter dem Begriff ITS-G5 **A** allokiert mit der Option, diese zukünftig zu erweitern. Die einzelnen Funkkanäle nutzen eine Bandbreite von 10 MHz. Der Kontrollkanal wird als **CCH** bezeichnet, die Servicekanäle als **SCH**.

Bei WAVE in den USA wurden die entsprechend 75 MHz allokierte Bandbreite in 7 Kanäle mit jeweils 10 MHz aufgeteilt. Die Nutzung desselben Frequenzbereichs in Europa und den USA ermöglicht eine einheitliche Hardware im Fahrzeug. (vgl. Abb. 8).[20]

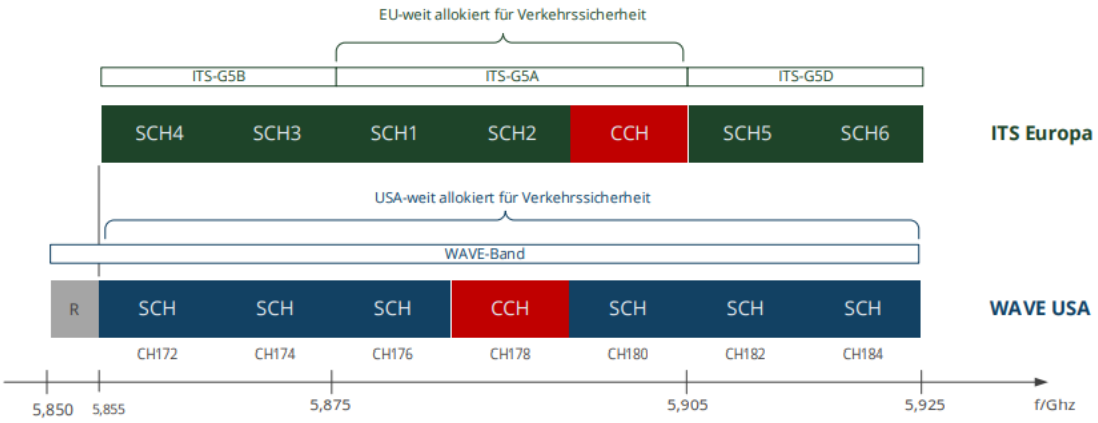


Abbildung 8: Frequenzbereiche für ITS-Applikation in Europa und USA [20]

Der Physical Layer wie auch der MAC-Layer basiert auf dem Wireless LAN Standard IEEE 802.11p, der den Anforderungen der mobilen Fahrzeugvernetzung genügt:

- Höhere Reichweiten (bis 1000 m)
- Hohe Geschwindigkeiten der Fahrzeuge
- Umfeld mit Mehrwegeausbreitung
- Parallele Ad-Hoc-Netzwerke
- Hohe Dienstgüte

Diese Anforderungen führten zur entsprechenden Spezifikation IEEE 802.11p:

- Entsprechend der Frequenzbänder 7 x 10 MHz → Halbierung der Datenrate gegenüber IEEE 802.11a (20 MHz)
- Verwendung des Orthogonalen Frequenzmultiplexverfahrens (OFDM). (vgl. Abb. 9)
    - ✓ 48 Datenträger und 4 Pilotträger
    - ✓ mit je 156,25 kHz Bandbreite
    - ✓ Symboldauer 8 µs
    - ✓ Guard-Intervall 1,6 µs → Verdopplung gegenüber IEEE 802.11a
- Modulationsverfahren der Subträger: BPSK, QPSK, 16-QAM, 64-QAM
- Maximale Übetragungsleistung 33 dBm (Vergleich 2,4 GHz WLAN/Bluetooth:20 dBm). [20]

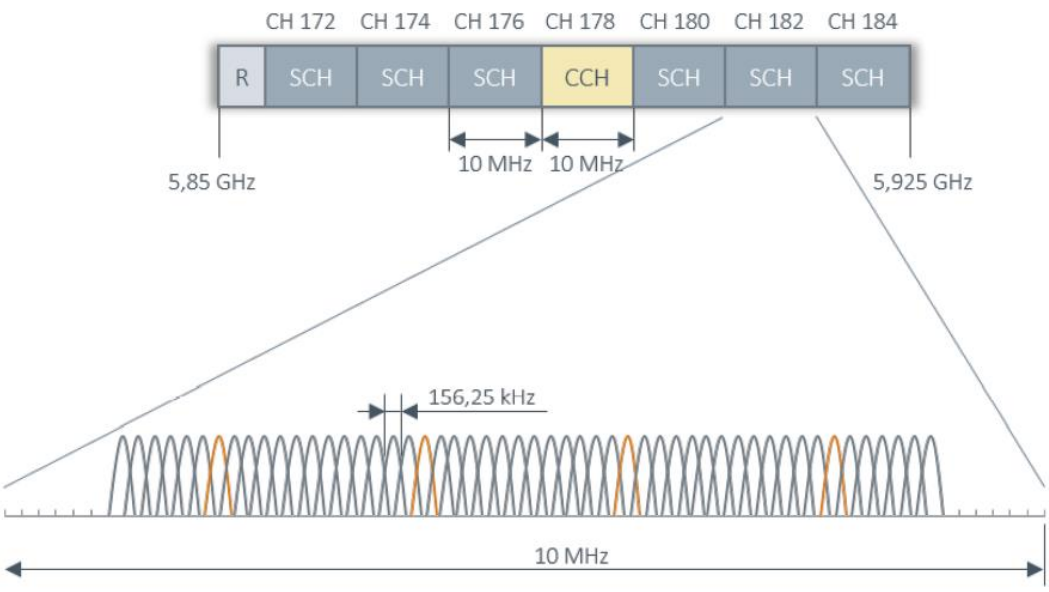


Abbildung 9: OFDM bei IEEE 802.11p[20]

In Abhängigkeit der genutzten Konfiguration ergeben sich acht verschiedene Transferraten (vgl. Abb. 10).

Mbit/s	Modulation	Coding	Bits/Subträger	Cod. Bits/Symbol	Datenbits/Symbol
3	BPSK	1/2	1	48	24
4,5	BPSK	3/4	1	48	36
6	QPSK	1/2	2	96	48
9	QPSK	3/4	2	96	72
12	16-QAM	1/2	4	192	96
18	16-QAM	3/4	4	192	144
24	64-QAM	2/3	6	288	192
27	64-QAM	3/4	6	288	216

Abbildung 10: Mögliche Transferraten bei IEEE 802.11p[20]

Abbildung 11 zeigt den Protokoll-Stack bei Verwendung des IEEE802.11p Standards.

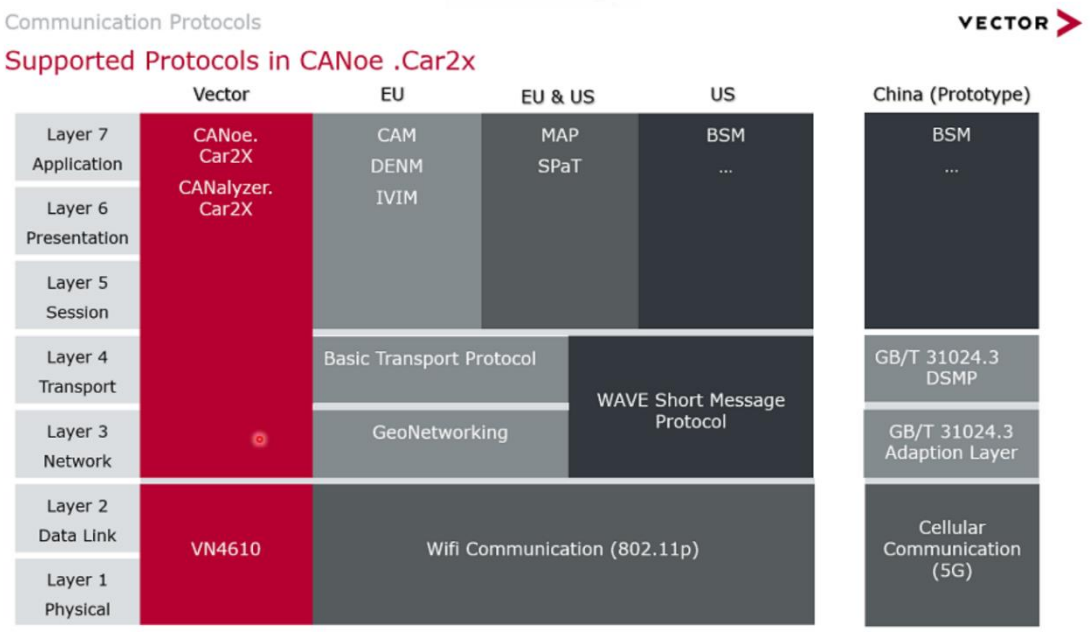


Abbildung 11: Unterstütztes Protokoll in V2X in Schichte 1 bis 7[Quelle: vector.de]

Auf Schichte 1 liegt IEEE 802.11p. IEEE 802.11p ist eine genehmigte Änderung des IEEE 802.11-Standards, um drahtlosen Zugriff in Fahrzeugumgebungen (WAVE), einem Fahrzeugkommunikationssystem, hinzuzufügen. Es definiert Verbesserungen auf 802.11 (die Basis der Produkte, die als Wi-Fi vermarktet werden), die zur Unterstützung von Anwendungen für intelligente Transportsysteme (INTELLIGENT Transportation Systems, ITS) erforderlich sind. Dazu gehört der Datenaustausch zwischen Hochgeschwindigkeitsfahrzeugen und zwischen den Fahrzeugen und der Straßeninfrastruktur, der sogenannten V2X-Kommunikation, im lizenzierten ITS-Band von 5,9 GHz (5,85–5,925 GHz). IEEE 1609 ist ein höherer Schichtstandard, der auf dem IEEE 802.11p basiert. Sie ist auch die Grundlage einer europäischen Norm für die Kommunikation mit Fahrzeugen, die als ETSI ITS-G5 bekannt ist.

802.11 WiFi wird für die Verwendung in der Fahrzeugkommunikation in Betracht gezogen. Es gibt zwei Hauptgründe, 802.11 WLANS gegenüber Mobilfunk- und WiMAX-Netzwerken zu bevorzugen. Erstens werden die WLAN-APs in den meisten entwickelten Städten der Welt massiv eingesetzt und bieten daher eine angemessene Infrastrukturunterstützung. Die bereits verfügbare WLAN-Infrastruktur macht hohe Investitionen für die Bereitstellung der Straßeninfrastruktur überflüssig. Da sie im freien und nicht lizenzierten ISM-Frequenzband (Industrial, Scientific and Medical) betrieben werden, entstehen ihnen keine zusätzlichen Kosten für dediziertes Spektrum, wie dies bei zellularen Systemen der Fall ist. Zweitens unterstützt WLANS Datenraten, die viel höher sind als bei WIMAX und Mobilfunknetzen. WLANS kann den schnellen

Informationsaustausch auch bei Fahrzeuggeschwindigkeiten unterstützen (Tufail et al. 2008). Mit zunehmendem Interesse an der Erkundung von & 802.11-Netzwerken in Fahrzeugumgebungen hat IEEE 802.11p WAVE standardisiert, um den Informationsaustausch zwischen Fahrzeugen sowie zwischen Fahrzeugen und Straßeninfrastruktur zu unterstützen.[8]

2.4 Aufbau der Botschaften

Auf Schichte 5-7 Sitzung werden Nachrichten wie die Cooperative Awareness Message (CAM) und die Decentralized Environmental Notification Messages (DENM) definiert. Die Informationen für diese Nachrichten werden in dieser Ebene generiert und zusammengestellt. Die Nachrichten werden mit Zeitstempeln und Positionsdaten versehen. In dieser Ebene wird auch geregelt, ob die Nachrichten periodisch oder ereignisbasiert generiert werden.

2.4.1 CAM

Bei den CAM-Messages handelt es sich um Statusinformationen über den Verkehrsfluss, die Fahrzeugposition, die Fahrgeschwindigkeit, die Fahrtrichtung, den Fahrzeugzustand und vieles mehr, die zwischen Fahrzeugen mittels Car-to-Car-Kommunikation (C2C) oder über Roadside Units (RSU) mittels Car-to-Infrastructure-Kommunikation (C2I) und der Verkehrsleitzentrale ausgetauscht werden. Die CAM-Meldungen werden einmal pro Sekunde ausgesendet.

Eingesetzt werden CAM-Nachrichten beispielsweise zyklisches Aussenden von Statusinformation wie Position, Geschwindigkeit. Weitere Nachrichten erfolgen als Basic Safety Messages (BMS).[9] (Siehe Tabelle 1)

Complete Message	Header	Signer_Info	
		Generation_Time	
		its_aid ITS-AID for CAM	
	CAM Information	Basis Container	ITS-Station Type
			Last Geographic Position
		High Frequency Container	Speed
			Driving Direction
			Longitudinal Acceleration
			Curvature
			Vehicle Length
			Vehicle Width
			Steering Angle
			Lane Number
			...
		Low Frequency Container	Vehicle Role
			Lights
			Trajectory
		Special Container	Emergency
			Police
			Fire Service
			Road Works
			Dangerous Goods
			Safety Car
			...
	Signature	ECDSA Signature of this Message	
	Certificate	According Certificate for Signature Verification	

Tabelle 1: Nachrichtenformat eines CAM [11].

Das CAM besteht aus einem Header, verschiedenen Datencontainern, z.B. dem Basiscontainer, einer Signatur und dem entsprechenden Zertifikat.

2.4.2 DENM

Eine DENM wird durch ein bestimmtes Ereignis ausgelöst und dient dazu andere Verkehrsteilnehmer zu benachrichtigen. Einige dieser Ereignisse wären zum Beispiel ein Stau, Glatteis auf der Straße, ein Unfall und ähnliche Ereignisse, welche das Verkehrsgeschehen beeinträchtigen.[10] (Siehe Tabelle 2)

Complete Message	Header	Signer_Info	
		Generation_Time	
		its_aid ITS-AID for DENM	
	DENM Information	Management Container	Last Vehicle Position (GPS)
			Event Identifier
			Time of Detection
			Time of Message Transmission
			Event Position (GPS)
			Validity Period
			Station Type (Motor Cycle, Vehicle, Truck)
			Message Update / Removal
			Relevant Local Message Area (geographic)
			Traffic Direction (forward, backwards, both)
			Transmission Interval
			....
		Situation Container	Information Quality (low -high, tbd)
			Event Type (Number)
			Linked Events
			Event Route (geographical)
		Location Container	Event Path
			Event Speed
			Event Direction
			Road Type
		A la carte Container	Road Works (Speed Limit, Lane Blockage....)
			....
	Signature	ECDSA Signature of this message	
	Certificate	According Certificate for Signature Verification	

Tabelle 2: Nachrichtenformat eines DENM [11]

Der DENM besteht aus einem Header, verschiedene Datencontainer, z. B. der Verwaltungscontainer, eine Signatur und das entsprechende Zertifikat.

2.4.3 SPAT

Eine SPAT (Signal Phase und Timing) beschreibt die aktuelle Phase an einer signalisierten Kreuzung, zusammen mit der Restzeit der Phase, für jede Spur (daher jede Neigung und Bewegung) der Kreuzung. Die Schätzung wird periodisch von der Kreuzung ausgestrahlt, sagt man einmal pro 100ms. [21] (Siehe Abb.12)

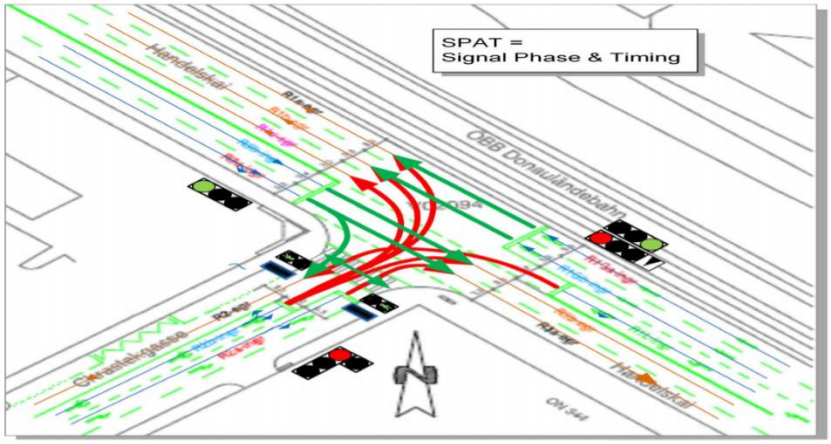


Abbildung 12: SPAT [22]

2.5 CANoe Software

2.5.1 Einführung

CANoe ist das vielseitige Softwarewerkzeug der Firma Vector Informatik für das Entwickeln und Testen einzelner Steuergeräte sowie ganzer Netzwerke. Zuverlässig unterstützt es Netzwerk-Designer, Entwicklungs- und Testingenieure im gesamten Entwicklungsprozess – von der Planung bis hin zum Systemtest.[14]

Kommende Fahrzeuggenerationen werden mehr und mehr in eine umfassende Dateninfrastruktur eingebunden sein, die die direkte Kommunikation zwischen Fahrzeugen (ITS Vehicle Station) einerseits und Infrastruktur (ITS Roadside Station) andererseits ermöglicht. Die Kommunikation erfolgt dabei über den WLAN-Standard IEEE 802.11p (ETSI ITS-G5 und IEEE 1609 WAVE). Die Option Car2x erweitert hierfür CANoe um WLAN-Kanäle nach IEEE 802.11p. Dies ermöglicht das direkte Analysieren sowohl der Car2x-spezifischen Netzwerk- und Transportprotokolle als auch der darauf aufsetzenden Applikationsnachrichten – im Umfeld Car2x sind das zum

Beispiel in Europa die Cooperative Awareness Message (CAM) und die Decentralized Environmental Notification Message (DENM) bzw. in den USA die Basic Safety Message (BSM). Dabei werden auch signierte Pakete (Secured Packets) unterstützt.[12] (Siehe Abb.13)

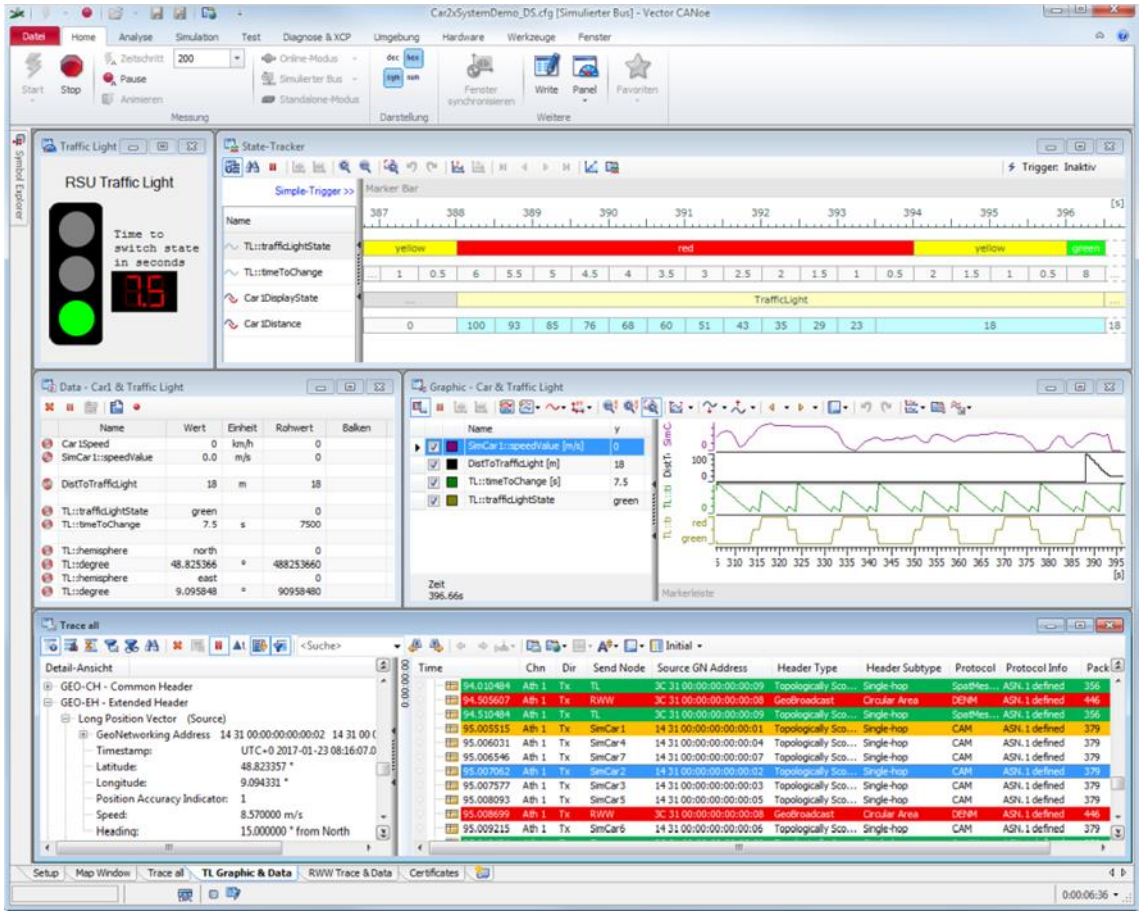


Abbildung 13: CANoe.Car2x simuliert Car2x-Ampel-Szenarien mit ITS Vehicle Stations. Im Trace-Fenster erfolgt die Analyse der Steuergeräte/Netzwerke [Quelle: Vector]

CAPL ist eine von Vector Informatik entwickelte Programmiersprache, die in den weit verbreiteten Software-Werkzeugen CANoe zur Verfügung steht.[14] Die in CAPL zur Verfügung stehende Car2x Funktionsbibliothek (Programmierschnittstelle) bietet spezielle Funktionen, um auf die Informationen (Signale/Daten) der empfangenen Pakete zugreifen und reagieren zu können. Diese Bibliothek ermöglicht es auch eine Umgebungssimulation erstellen zu können, die als Voraussetzung für die Stimulation von Steuergeräten notwendig ist, damit diese gezielt getestet werden. Für die Darstellung der Informationen sorgen neben einem speziellen Karten-Fenster die typischen CANoe Analyseblöcke wie das Trace-, Daten- und Grafik-Fenster.

2.5.2 CANoe-Features

Zu den Grundfunktionen von CANoe gehören:

1.Einsatz von Datenbasen, die das jeweilige Netzwerk beschreiben

2.Simulation kompletter Systeme und Restbussimulationen

3.Analyse der Buskommunikation

4.Test kompletter Netze und/oder einzelner Steuergeräte

5.Diagnosekommunikation nach KWP2000 und UDS sowie Einsatz als vollwertiger Diagnosetester

6.Freie Programmierbarkeit durch die Programmiersprache CAPL zur Unterstützung von Simulation, Analyse und Test

7.Erstellen benutzerdefinierter Oberflächen zur Steuerung der Simulation und Tests oder zur Anzeige der Analysedaten

8.Einbinden von zusätzlicher I/O-Hardware und/oder spezieller Test-Hardware (VT System)

9.Intuitive Benutzerschnittstelle mit flexiblem Docking-Konzept und übersichtlichen Menüstrukturen

10.Unterstützung neuer Vector Bus-Hardware

2.5.3 CANoe-Programmoberfläche

Wenn CANoe das erste Mal öffnen, dann wird automatisch eine neue Konfiguration mit den Desktops Trace, Configuration und Analysis geöffnet. (Siehe Abb.14)

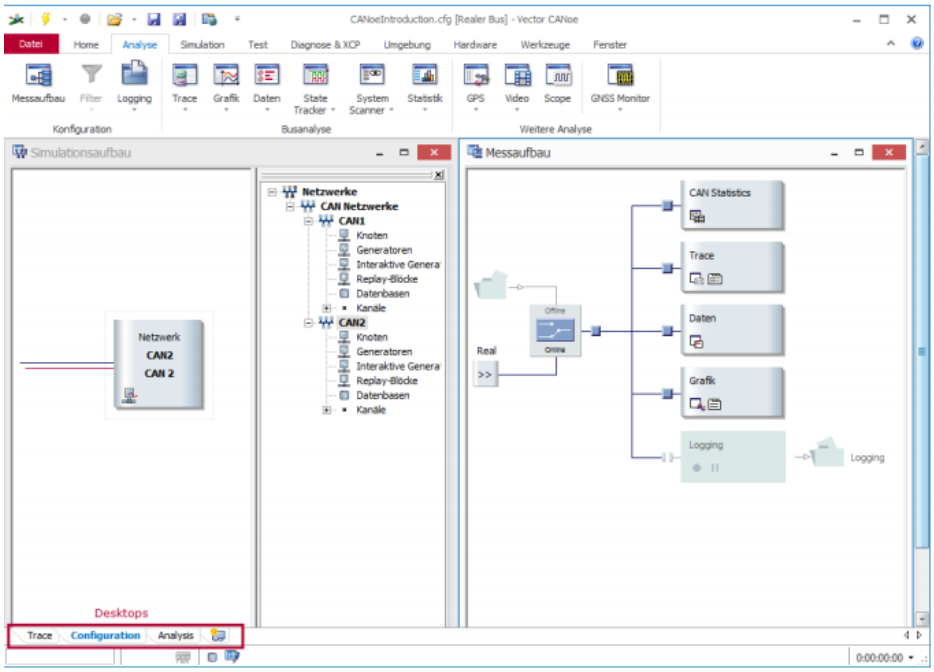


Abbildung 14: Beispielkonfiguration für 2 CAN Kanäle [Quelle: Vector.com]

CANoe verfügt über verschiedene Analysefenster (z.B. Trace-, Daten-, Grafik- und Statistik-Fenster) sowie einen Messaufbau und einen Simulationsaufbau, die Ihnen den Datenfluss anzeigen und über die man CANoe gleichzeitig konfigurieren kann. Man erreicht alle Fenster des Programms über das Menüband.

Die Informationen die in jedem Auswerteblock eintreffen, werden im zugehörigen Analysefenster dargestellt. So stellt z.B. das Trace-Fenster alle Informationen dar, die im Trace-Block ankommen, während das Grafik-Fenster die Informationen anzeigt, die im Grafik-Block eintreffen. Einzige Ausnahme ist der Logging-Block, dem kein Fenster, sondern eine Datei zugeordnet ist, in der die am Block eintreffenden Daten aufgezeichnet werden.[13] (Siehe Abb.15)

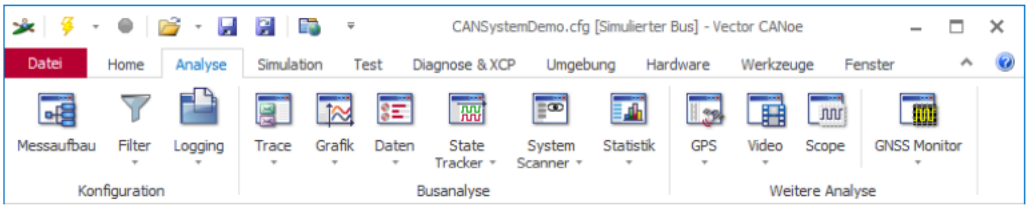


Abbildung 15: Menüband, Registerkarte Analyse

2.5.4 Simulationsaufbau

Im Simulationsaufbau (Desktop Configuration oder Registerkarte Simulation des Mübands|Simulationsaufbau) wird das Gesamtsystem mit dem CAN-Bus und allen Netzwerkknoten grafisch dargestellt. Der simulierte Bus wird dabei durch eine rote horizontale Linie repräsentiert. Die darüber liegende blaue Linie symbolisiert den realen Bus. Beide Busse sind über die Interface-Hardware miteinander verbunden. Um Daten aus CANoe auf den Bus zu senden, fügt man im Simulationsaufbau über Kontextmenü des Busstrangs Sendeblöcke ein, die mit der roten Linie verbunden werden müssen. (Siehe Abb.13)

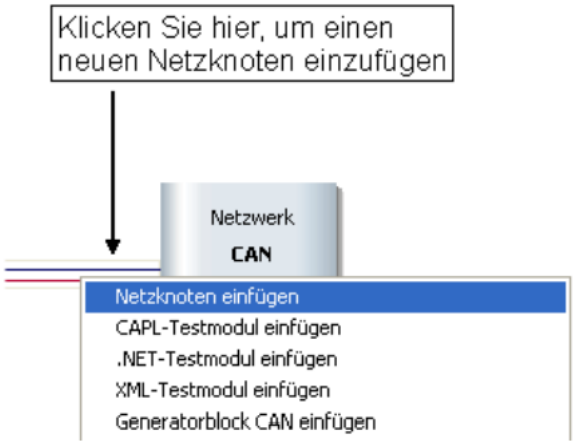


Abbildung 16: Bussymbol im Simulationsaufbau mit Kontextmenü des Busstrangs [Quelle:vector.com]

2.5.5 Messaufbau

Das Datenflussdiagramm des CANoe Messaufbaus (Desktop Configuration oder Registerkarte Analyse des Menübands|Messaufbau) enthält links die Verbindung zum Simulationsaufbau – symbolisiert durch das >> Symbol – und rechts verschiedene Auswerteblöcke als Datensenken. Die Daten fließen also von links nach rechts. Zur Veranschaulichung des Datenflusses sind zwischen den einzelnen Elementen Verbindungsleitungen und Verzweigungen eingezeichnet.

Im Datenflussdiagramm erkennt man ferner kleine Quadrate. An diesen Einfügepunkten (Hot Spots) kann man weitere Funktionsblöcke zur Manipulation des Datenflusses (Filter, Replay-Block, CAPLProgrammblock mit benutzerdefinierbaren Funktionen) einfügen.[13] (Siehe Abb.17)

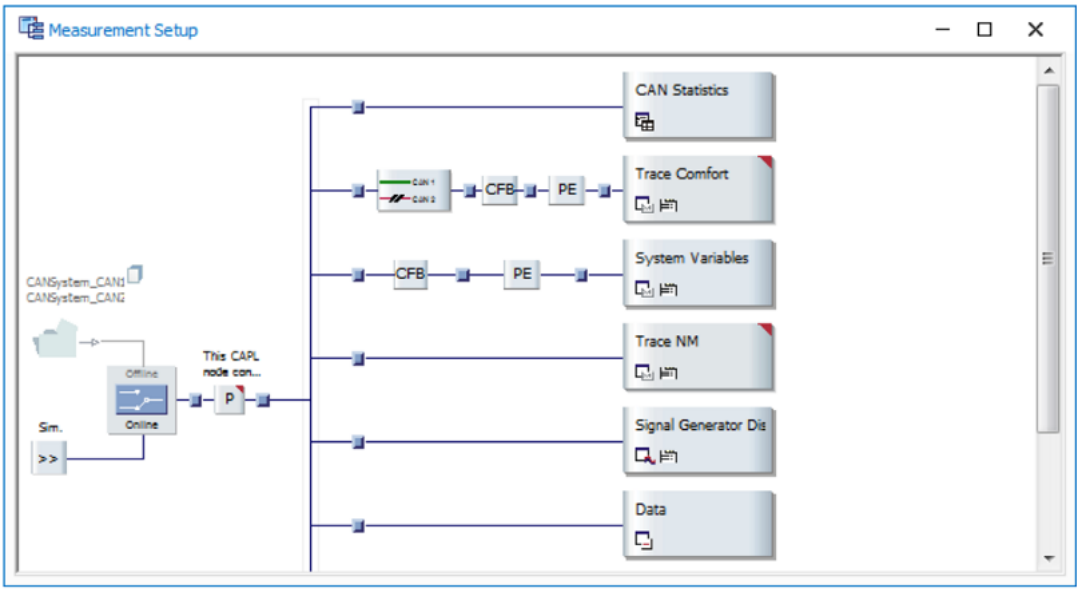


Abbildung 17: Messaufbau

2.6 VN4610

2.6.1 Begriff

Das VN4610 ist ein leistungsfähiges Interface der Firma Vector Informatik mit USB-Schnittstelle für den Zugriff auf IEEE 802.11p- und CAN(FD) -Netzwerke. Die IEEE 802.11p basierte Dedicated Short Range Communication (DSRC) kommuniziert im 5.9 GHz Bereich. Das VN4610 unterstützt das ungefilterte Empfangen und Senden von IEEE 802.11p Frames, die zur Umsetzung von Car2x/V2x Applikationen verwendet werden. Die empfangenen IEEE 802.11p Funkframes werden zeitsynchron zu den CAN(FD) Nachrichten an die Applikation übergeben. Der eingebaute GNSS-Empfänger liefert dabei die GNSS-Zeit sowie die aktuelle GNSS-Position. (Siehe Abb.18)

Abbildung 18: VN4610 Netzwerk Interface mit GNSS Empfänger [Quelle: Vector.com]

2.6.2 Vorteile im Überblick

1.Senden/Empfangen von Frames nach IEEE 802.11p.

2.Zwei konfigurierbare IEEE 802.11p WLAN Funkkanäle.

3.Ungefiltertes Weiterleiten von IEEE 802.11p Datenpaketen an die Applikation.

4.Einstellbare Kommunikationsparameter wie Funkkanalauswahl, Bandbreite, Sendeleistung, Modulationsart und Protokollformat LPD/EPD.

5.Zwei CAN-Highspeed Kanäle CAN(FD) fähig.

6.GNSS Empfänger liefert aktuelle Position und Zeit.

7.Präzise Zeitstempel (Genauigkeit 1µs) basierend auf GNSS Zeit.

8.VN4610 und CANoe.Car2x/CANalyzer.Car2x sind optimal aufeinander abgestimmt.

9.Synchronisation mit mehreren Interfaces und mit anderen Bussystemen (Ethernet, CAN, LIN...).

10.Robustes Gehäuse, Stromversorgung und Temperaturbereich ideal für Automotive- sowie industrielle Anwendungen.

2.6.3 Anwendungsgebiete

Das VN4610 erfüllt alle hardwaretechnischen Anforderungen, die als Grundlage zum Testen von DSRC-Applikationen über IEEE 802.11p Funkkanäle benötigt werden.

Analyse: Das VN4610 leitet für die Analyse alle empfangen Funkframes der beiden Funkkanäle ungefiltert an das Testwerkzeug weiter. Somit können auch Frames analysiert werden, die bei einem Steuergerät aufgrund des Timings, der Geo-Informationen oder von Car2x/V2x Protokollfehlern verworfen würden. Da die Zeitstempel der Nachrichten auf den Buskanälen zeitlich synchronisiert sind, können zusätzlich auch Latenzmessungen durchgeführt werden.

Simulation/Stimulation: CANoe.Car2x zusammen mit dem VN4610 bietet eine perfekt aufeinander abgestimmte Lösung zur Erstellung einer Umgebungsstimulation zum Testen von Car2x/V2x Applikationen. Das VN4610 sendet dabei die übertragenen Frames, wobei die Kommunikationsparameter einfach und individuell für die unterschiedlichen Tests konfiguriert werden können.

GNSS-Empfänger: Das VN4610 liefert präzise Positions-, Zeit- und Geschwindigkeitsinformationen die der Applikation bspw. als Teststimulus oder zur Dokumentation dienen können. Darüber hinaus können die absoluten GNSS-Zeitstempel zur Synchronisation der Aufzeichnungen von verteilten Messungen bei der anschließenden Analyse verwendet werden.[14] (Siehe Abb.19)

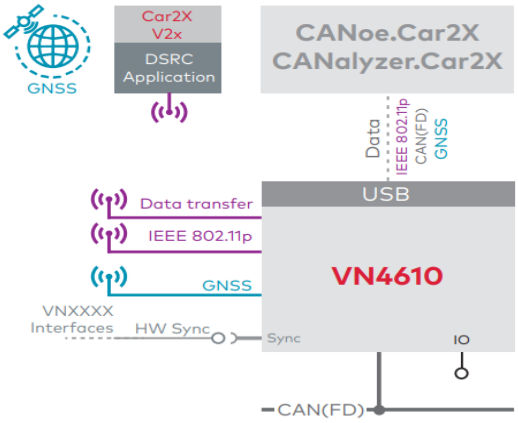


Abbildung 19: Beschaltungsmöglichkeiten und Anwendungsfälle [Quelle: Vector.com]

2.7 CAPL Programmierung

Die Programmiersprache CAPL basiert auf C Programmierung. Diejenigen, die mit C Programmierung vertraut sind, benötigen minimale Lernzeit, um mit der Verwendung zu beginnen CAPL effektiv. Die Syntax, die meisten Standardfunktionen und einige mathematische Funktionen von ANSI C wurden alle in CAPL integriert, was die C-Basis um zusätzliche netzwerkspezifische Funktionen und Datentypen erweitert.[15] (Siehe Abb.20)

Der CAPL Browser ist mehr als ein Editor für CAPL-Programme. Er bietet die Funktionen einer modernen Entwicklungsumgebung, wie:

1.Code-Ergänzung und Syntaxprüfung während des Schreibens.

2.Konfigurierbares Syntax-Highlighting.

3.Syntaxsensitive Einrückung.

4.Einklappbare Funktionsblöcke und Funktionsreferenz in einer Baumansicht zur schnelleren Navigation.

5.Suchen und Ersetzen in einzelnen oder mehreren Dateien > Hilfe mit Referenzen auf Funktionen.

6.Aufruf des Compilers mit direkt anwählbaren Quelltextzeilen im Fehlerfall.

7.Hierarchische Funktionsliste mit Suchfunktion zur direkten Übernahme in den Darüber hinaus stehen im CAPL Browser die Objekte aus der CANoe Datenbasis zur

Verfügung. Diese werden ebenfalls in einer Baumansicht dargestellt. Im sogenannten Symbol Explorer kann auf die folgenden Aspekte der Datenbasis zugegriffen werden:

8.Netzwerksymbole, wie Knoten, Botschaften und Signale.

9.Umgebungsdaten, das sind datenbasisspezifische Umgebungsvariablen und die CANoe-weit verwendeten Systemvariablen.

10.Alle Diagnosesymbole, wie Requests, Responses und Fehlerspeicher.[12]

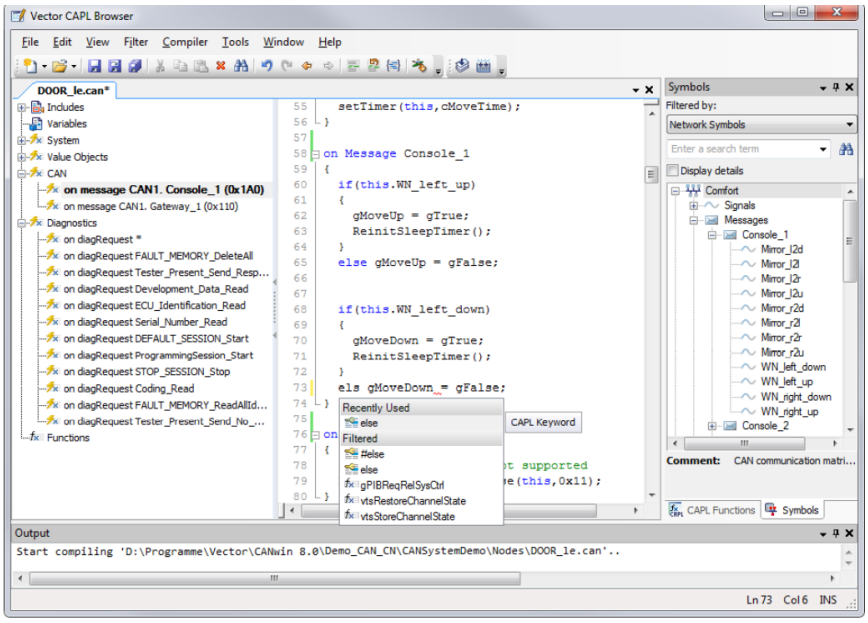


Abbildung 20: CAPL Browser mit geöffnetem CAPL-Programm, seinen enthaltenen Ereignisprozeduren und in der Datenbasis enthaltenen Netzwerksymbolen [Quelle: vector.com]

Für die Fehlersuche in CAPL- und .NET-Programmen steht der Vector Debugger zur Verfügung. Mit ihm können im Quelltext der Programme Haltepunkte eingefügt und Werte von Variablen geprüft werden. Im simulierten Modus ist ein Debuggen aller CAPL- und .NET-Programme möglich, da die Simulation zu diesem Zweck angehalten wird. Bei der Verwendung von realer Hardware ist ein Debuggen nur in Programmen von Testmodulen möglich, da die von der Hardware gesendeten Ereignisse weiterhin ausgewertet werden müssen. [12] (Siehe Abb.21)

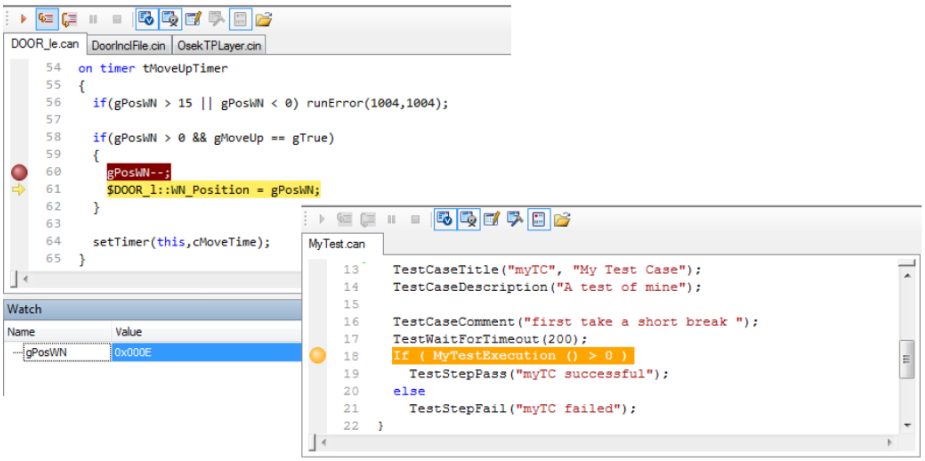


Abbildung 21: Debugger mit Haltepunkten [Quelle: vector.com]

3 Erstellung von Praktikumsversuchen

Um die in den Vorlesungen vermittele Theorie zu V2X-Technologien den Stduenten näher zu bringen, soll ein entsprechendes Praktikum Konzipiert werden. Im Folgenden werden dafür 3 Versuche vorgestellt. Der erste Versuch geht es um Beaconing und zweiter Versuch handelt es sich um Emergency situation. Außerdem ist dritter Versuch relevant über die Kommunikation mit RSU.

3.1 Versuch1-Beaconing

3.1.1 Ziel des Versuchs 1

Die Fahrzeuge kommunizieren miteinander zum zyklischen Senden von Position, Fahrrichtung, Geschwindigkeit. In diesem Versuch soll die Kommunikation zwischen 2 Fahrzeuge simuliert werden.

Die Aufgabe des Versuch1s:

Student A:

- Definition einer speziellen Route im „Szenario Manager“ als Endlos-Schleife (z.B. in Chemnitz oder Mittweida)
- Das Fahrzeug soll auf der Route unterschiedliche Geschwindigkeiten haben
- Diese CAM-Botschaften des Student B beinhalten die Positionen von Student B à das Fahrzeug von Student B soll damit zusätzlich zum eigenen Fahrzeug im Map-Window angezeigt werden
- Die aktuellen Positionen von Student A und B sowie die aktuellen Geschwindigkeiten sollen in einem eigenen Fenster dargestellt werden (z.B. Dashboard)

Student B:

- Definition einer speziellen Route im „Szenario Manager“ als Endlos-Schleife (z.B. in Chemnitz oder Mittweida) (Andere Route als Student A)
- Das Fahrzeug soll auf der Route unterschiedliche Geschwindigkeiten haben

- Diese CAM-Botschaften des Student A beinhalten die Positionen von Student A das Fahrzeug von Student A soll damit zusätzlich zum eigenen Fahrzeug im Map-Window angezeigt werden

- Die aktuellen Positionen von Student A und B sowie die aktuellen Geschwindigkeiten sollen in einem eigenen Fenster dargestellt werden (z.B. Dashboard)

Hier ist die Skizze der Versuch1: (Siehe Abb. 22)

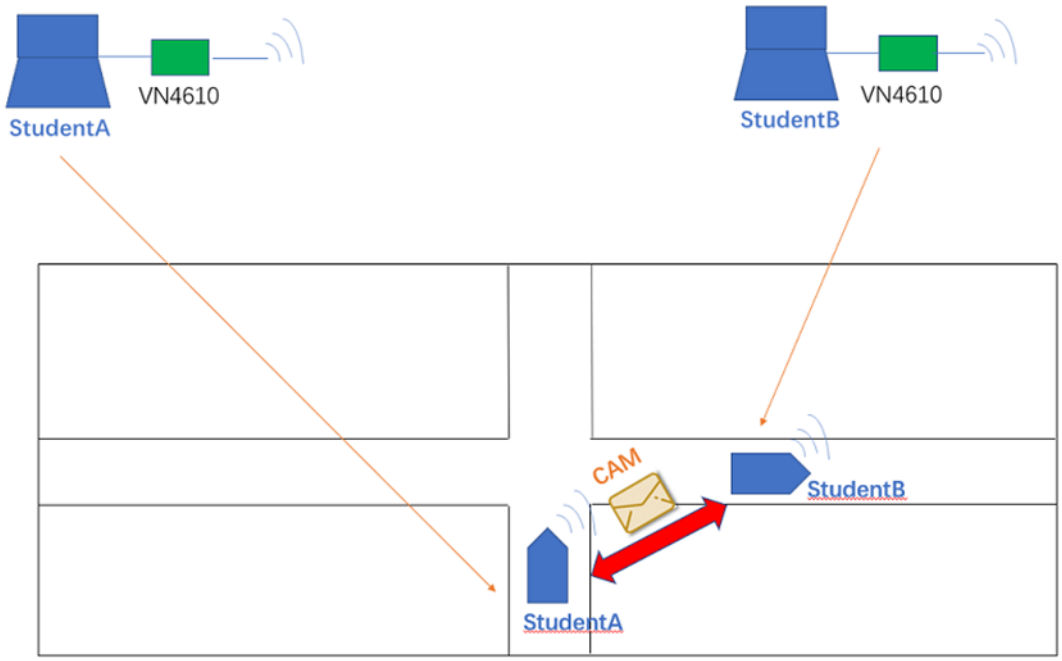


Abbildung 22: Skizze der Versuch1

3.1.2 Projekt erstellen

Zunächst wird eine neue Car2x-Konfiguration erstellt.

Dann wird „EU_ApplMsg.xml“ im Simulationsaufbau eine Car2x-Datenbasis hinzugefügt. Die Informationen dieser Datenbasen können in CANoe symbolisch dargestellt und verwendet werden. Botschaften können in einer Sendeliste manuell oder über eine Datenbasis konfiguriert werden.[14] Datenbasis beinhaltet CAM-Botschaft, DENM-Botschaft.

1 Knoten wird angelegt (abhängig von Student A oder B). Die Datenbasis sollte Knoten beinhalten, welche der Station im Szenarien Editor entsprechen.

- ✓ StudentA: Das zu testende Fahrzeug der StudentA.

✓ StudentB: Das zu testende Fahrzeug der StudentB.

Konfigurieren die Netzwerknote und Tx Nachrichten.

Um die Knoten auch im Simulationsaufbau anzuzeigen, werden die Node Sychronization (Knoten Sychronisierung) verwendet. Wählen hierzu Node Sychronization über das Kontextmenü der hinzugefügten Datenbasis aus. Der Knote wird anschließend in den Simulationsaufbau eingefügt. (Siehe Abb. 23,24)

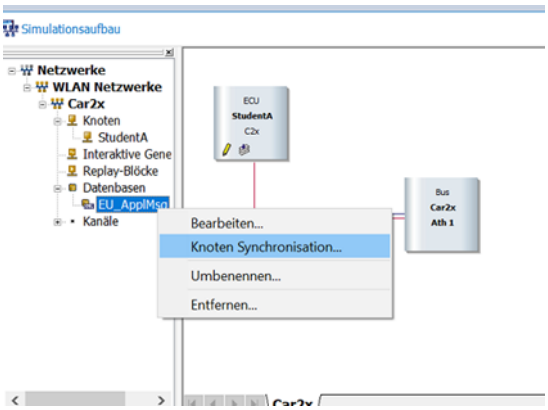


Abbildung 23: Knote Synchronisation auf PC1

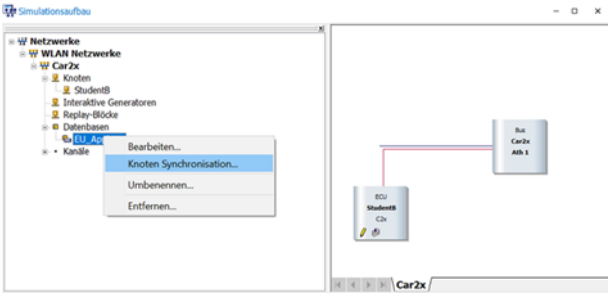


Abbildung 24: Knote Synchronisation auf PC2

3.1.3 Szenario erstellen

Der Car2x Scenario Editor über die Menüleiste in CANoe wird geöffnet, um Scenario zu erstellen.

Im Screenshot rechts wurde eine Route mit 1 Station im Szenarien Editor erstellt. Die Namen der Station müssen mit den Namen der Knoten in der Datenbasis

übereinstimmen. Die Farbe der Route und Station im Eigenschaftenfenster kann man selbst bestimmen, um klar zu sehen. Geben der Station unterschiedliche ID.

Beispiel für Route 1(Student A):

Route1 beginnt auf der Georgstraße auf der Chemnitzkarte. Fahren etwa 119 Meter geradeaus und biegen rechts ab, um die Str. Nationen zu erreichen. Gehen 108 Meter die Str. Nationen entlang und biegen dann rechts ab, um Minna-Simon-Straße zu betreten. Gehen dann 116 Meter die Minna-Simon-Straße entlang und biegen auch rechts in die Mauerstraße ab. Schließlich kehrt man zum Ursprung zurück. (Siehe Abb. 25)

Abbildung 25: Eine Route auf PC1

Beispiel für Route 2(Student B):

Route2 beginnt auf der Karl-Liebknecht-straße auf der Chemnitzkarte. Fahren etwa 151 Meter geradeaus und biegen rechts ab, um die Georgstraße zu erreichen. Gehen 169 Meter die Georgstraße entlang und biegen dann rechts ab, um Str. Nationen zu betreten. Gehen dann 157 Meter die Str. Nationen entlang und biegen auch rechts in die Richard-Tauber-Straße ab. Schließlich kehrt man zum Ursprung zurück. (Siehe Abb. 26)

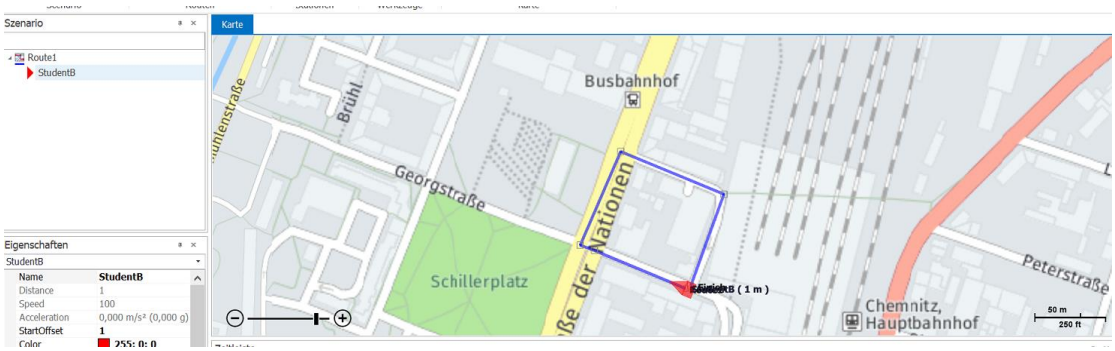


Abbildung 26: Eine Route auf PC2

Für die Station werden in der Zeitleiste des Car2x Szenarien Editors Attribute für die Geschwindigkeiten der Fahrzeuge angelegt. Wenn das Fahrzeug des Studenten A oder Studenten B abbiegen möchte, werden die Geschwindigkeiten abbremsen, um die Geschwindigkeiten zu reduzieren.

Das erstellte Szenario wird über den „Szenarien Manager“ in CANoe zur Konfiguration hinzugefügt. Der Szenarien Manager wird über die Toolbar in CANoe unter dem Tab Umgebung (Environment) aufgerufen.

Entsprechen die Namen der Station des Szenarios denen der Knoten in der Datenbank, werden diese im Szenarien Manager mit einem orangenen Icon angezeigt.

3.1.4 Programmierung

Systemvariablen werden häufig von CANoe-Komponenten verwendet. Meistens werden automatisch generiert und können nicht bearbeitet werden. Sie gehören zu einem definierten Namespace.[16] Vor Programmierung sollte Systemvariablen konfiguriert werden. (Siehe Tabelle 3)

StudentA::distance	Der Abstand zwischen StudentA und StudentB
StudentA::latitude	Der Breitengrad des Studenten A. Es beschreibt die genaue Position des Fahrzeugs.
StudentA::longtitude	Geographische Länge des Studenten A. Es beschreibt die genaue Position des Fahrzeugs.
StudentA::heading	Tatsächliche Fahrt richtungsbezogen. Es beschreibt die genaue Position des Fahrzeugs.
StudentA::Speed	Die Geschwindigkeit des Studenten A.
StudentB::distance	Der Abstand zwischen StudentA und StudentB
StudentB::latitude	Der Breitengrad des Studenten B. Es beschreibt die genaue Position des Fahrzeugs.
StudentB::longtitude	Geographische Länge des Studenten B. Es beschreibt die genaue Position des Fahrzeugs.

StudentB::heading	Tatsächliche Fahrt richtungsbezogen. Es beschreibt die genaue Position des Fahrzeugs.
StudentB::Speed	Die Geschwindigkeit des Studenten B.

Tabelle 3: Die Definition der Systemvariablen in versuch 1

Abbildung 27,28 zeigt die Definition der Latitude, Longtitude, Heading.

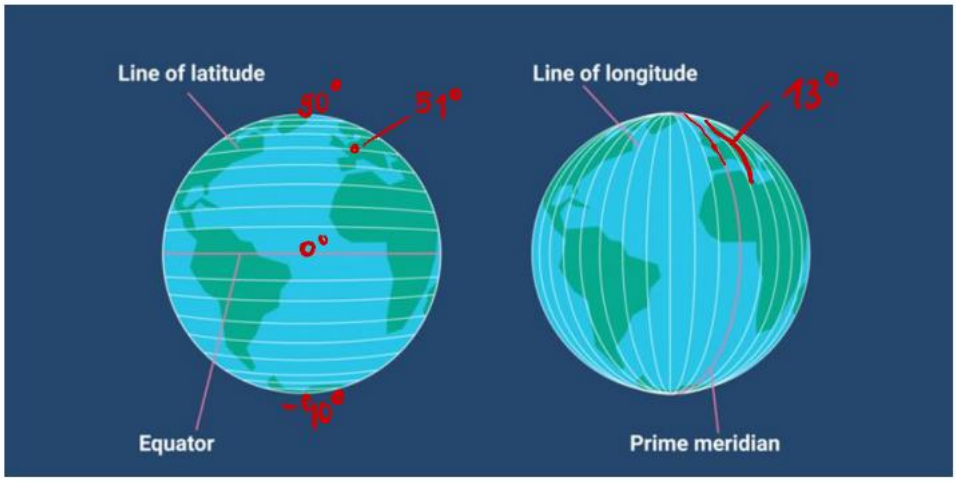


Abbildung 27: Latitude, Longtitude [Quelle:timeanddate.com]

Beispiele:

- Chemnitz:

  Latitude: 50°83'33.32"N, Longtitude: 12° 55' 0 E.

- London:

  Latitude: 51°30'30.71"N, Longtitude: 0°

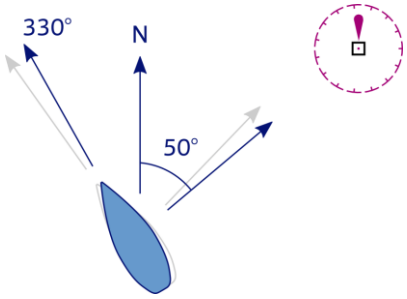


Abbildung 28: Heading

Definition einer speziellen Route im „Szenario Manager“ als Endlos-Schleife. Aus diesem Ziel programmiert man durch CAPL. Hier ist Programmablaufplan. (Siehe Abb. 29)

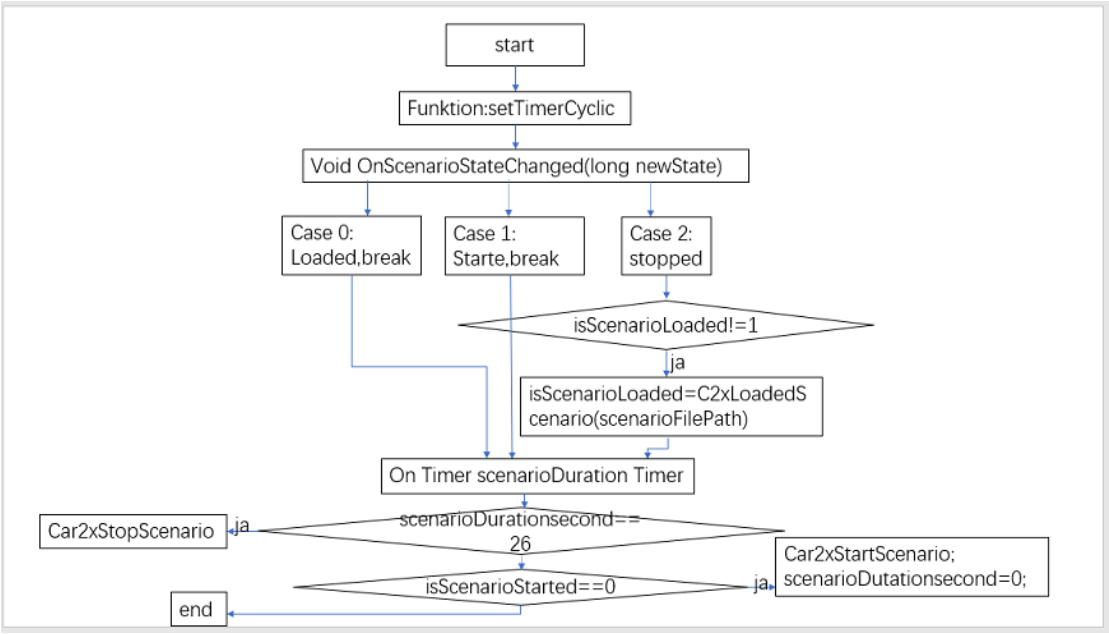


Abbildung 29: Programmablaufplan

Nachdem man Systemvariable definiert hat, kann man Botschaft einrichten, um Position, Geschwindigkeiten und Abstand zu senden.

Nicht nur einrichtet man die Fahrzeuge als endlose Schleife, sondern Botschaft auch endlose ist. Folglich sollte Sequenz periodisch ausführen.

Um Abstand zwischen den beiden Fahrzeugen zu berechnen, muss man eine Regel verwenden. (Siehe Abb. 30)

LatA bedeutet StudentA::latitude, LatB bedeutet StudentB::latitude

LonA bedeutet StudentA::longtitude, LonB bedeutet StudentB::longtitude

Δa bedeutet Absolutwert der Latitude zwischen StudentA und StudentB

Δb bedeutet Absolutwert der Longtitude zwischen StudentA und StudentB

Δc bedeutet Absolutwert der Abstände zwischen StudentA und StudentB

r bedeutet Erdradius.

$$\Delta a = |LatA - LatB| \cdot \frac{2\pi}{360} \ [rad]$$

$$\Delta b = |LonB - LonA| \cdot \frac{2\pi}{360} \ [rad]$$

Nach der Formel: haversine formular [17]:

$$\Delta c = 2r \cdot \arcsin\left(\sqrt{sin^2(\frac{\Delta a}{2}) + \cos(LatA) \cdot \cos(LatB) \cdot sin^2(\frac{\Delta b}{2})}\right) \ [km]$$

Die Formel ist nur eine Annäherung, wenn sie auf die Erde angewendet wird, was keine perfekte Kugel ist: Der "Erdradius" r variiert von 6356.752 km an den Polen bis 6378.137 km am Äquator. Beispiel wie die Abbildung 30 zeigt die Position der Fahrzeuge in Chemnitz.

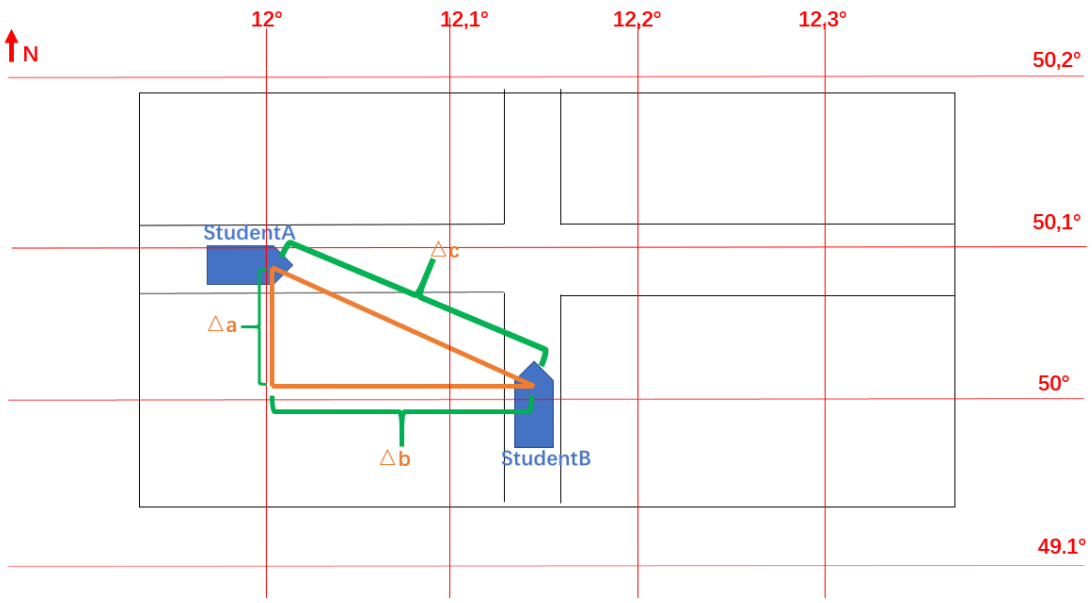


Abbildung 30: Abstand berechnen

Es dauert 26 Sekunden, bis StudentA und StudentB eine Runde absolviert haben. Hier sind Speed, Latitude, Longtitude, Distance, Heading des Studenten A in 0. Sekunde durch Botschaft1 und des Studenten B in 0. Sekunde durch Botschaft2.(Siehe Abb. 31,32)

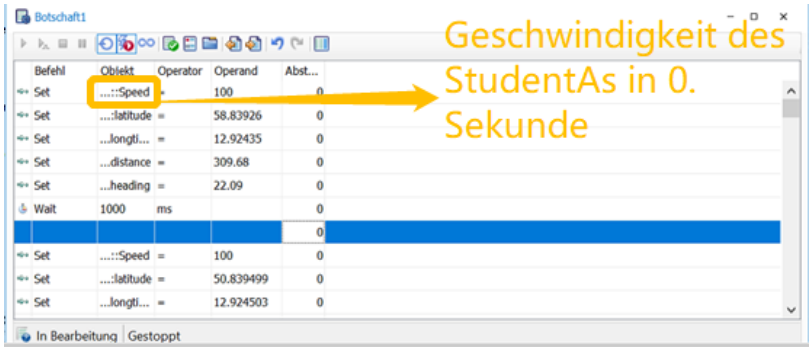


Abbildung 31: Botschaft1

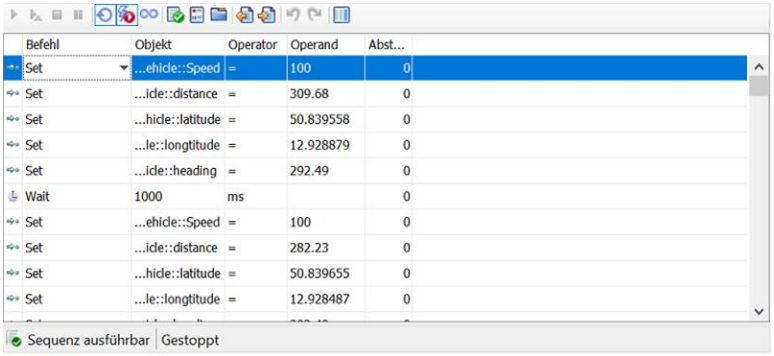


Abbildung 32: Botschaft2

3.1.5 Zertifikate

Für die gesicherte Car2x-Kommunikation wird ein Public Key Verfahren verwendet. Dafür sind Zertifikate nötig, die unter anderem die öffentlichen Schlüssel enthalten. Die dazugehörigen privaten Schlüssel werden im Zertifikatmanager verwaltet.

Bei Car2x / V2x bestätigt ein Zertifikat, dass der Eigentümer des Zertifikats die im Zertifikat enthaltenen Schlüssel und Rechte besitzt. Dazu enthält das Zertifikat neben den dafür notwendigen Daten auch die Identität des Zertifikatausstellers sowie dessen digitale Unterschrift, mit der die Korrektheit des Zertifikats verifiziert werden kann.

Der Zertifikataussteller weist sich selbst ebenfalls durch ein Zertifikat aus, das von ihm selbst oder von einem weiteren Aussteller unterschrieben ist, der sich wiederum durch ein Zertifikat ausweist.

Damit sich zwei ITS Stations, die sich nicht kennen, trotzdem gegenseitig vertrauen können, müssen diese ihre Zertifikate austauschen und den jeweiligen Zertifikatsausstellern vertrauen.[14]

Im globalen Dialog Optionen werden zentrale Einstellungen für die geladene Konfiguration und für das Programm vornehmen. (Siehe Abb. 33)

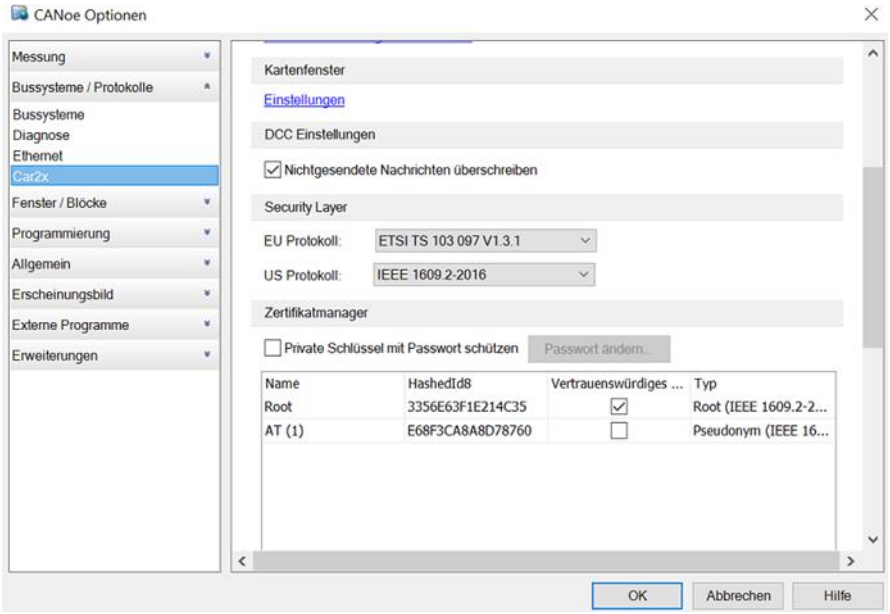


Abbildung 33: Zertifikate fertigen auf PC1 und PC2

Und Tabelle 4 zeigt die Eigenschaft der Zertifikate.

Subjekt Typ	Der Subjekt Typ legt den Verwendungszweck des Zertifikats fest. Root Zertifikate können immer erzeugt werden. Alle anderen Zertifikatarten können nur dann erzeugt werden, wenn ein Elternzertifikat mit passendem privatem Schlüssel vorhanden ist.
Subjekt Name	Der Subjekt Name identifiziert bei CA Zertifikaten den Zertifikatinhaber. Bei Pseudonym Zertifikaten bleibt der Name leer.
HashedId8	der Digest des Zertifikats.
Vertrauenswürdiges Zertifikat	Angezeigt werden ein optional zugeordneter Klartextname, der Digest (HashedId8) des Zertifikats und ob das Zertifikat als vertrauenswürdig eingestuft werden soll. Zertifikate, die nicht zur eingestellten Version des Standards passen, werden grau dargestellt und werden während der Messung nicht berücksichtigt.

	Wenn man das Kontrollfeld Vertrauenswürdiges Zertifikat aktiviert, wird diesem Zertifikat und allen mit ihm korrekt signierten Zertifikaten bzw. Botschaften vertraut. Im Allgemeinen ist dies für Root-Zertifikate der Fall.[14]

Tabelle 4: Die Eigenschaft der Zertifikate

3.1.6 Panel

Die aktuellen Positionen von Student A und B sowie die aktuellen Geschwindigkeiten sollen in einem eigenen Fenster dargestellt werden (z.B. Dashboard)

Daher Siehe man drei Bereich. (Siehe Abb. 34)

1: Es gibt definiertes Symbol.

2: Den Bereich, auf dem Panel Control Botton liegt. Man kann sich die Größe des Bereiches und Panel Control Buttons ändern.

3: Panel Control Button List.

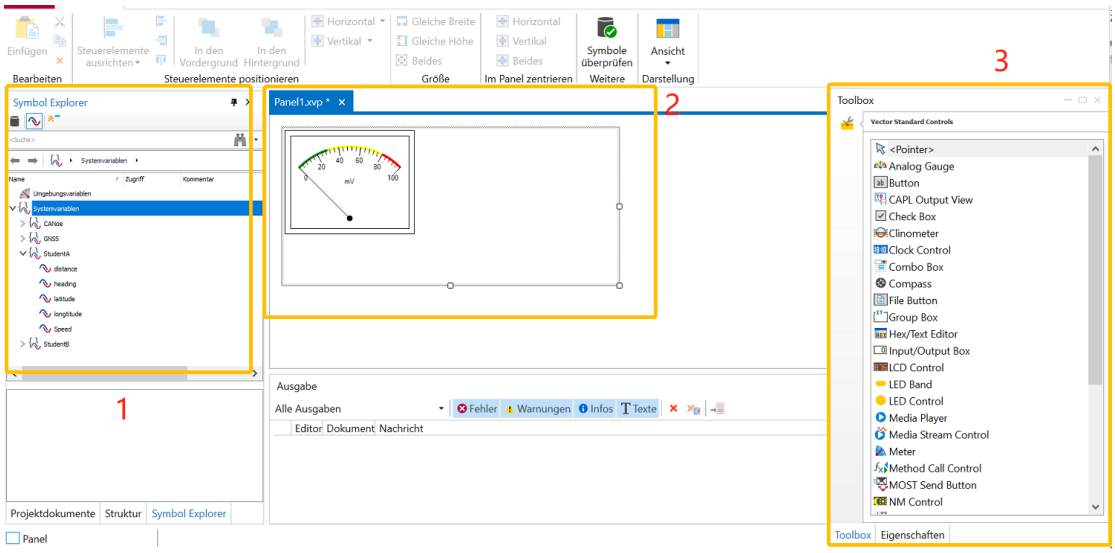


Abbildung 34: Drei Bereich im Panel Design

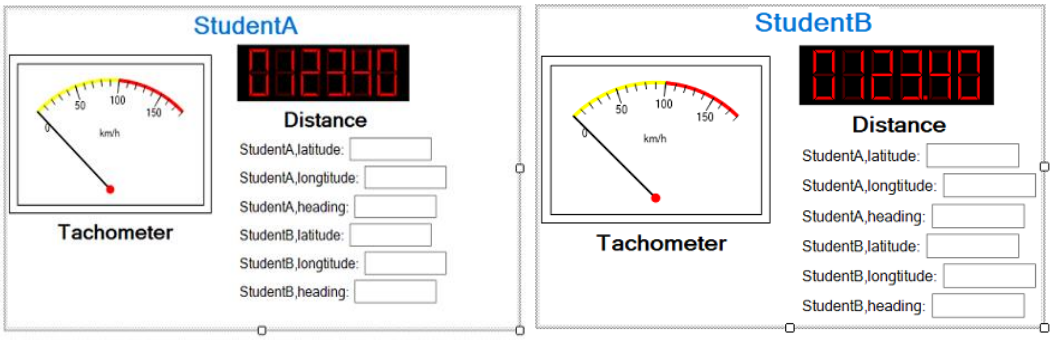


Abbildung 35: Panel der StudentA und StudentB

Abbildung 35 zeigt Dashboard des Studenten A und Studenten B.

3.2 Versuch2-Senden von Gefahren-Botschaften

3.2.1 Ziel des Versuchs 2

Zwei Fahrzeuge kommunizieren miteinander zum Senden von Position, Fahrrichtung, Geschwindigkeit und Gefahren-Botschaften. In diesem Versuch soll die Kommunikation zwischen 2 Fahrzeuge simuliert werden.

Student A:

- Definition einer speziellen Route im „Szenario Manager“ (z.B. in Chemnitz oder Mittweida)
- Das Fahrzeug soll auf der Route unterschiedliche Geschwindigkeiten haben
- Diese CAM-Botschaften des Student B beinhalten die Positionen von Student B das Fahrzeug von Student B soll damit zusätzlich zum eigenen Fahrzeug im Map-Window angezeigt werden
- Die aktuellen Positionen von Student A und B sowie die aktuellen Geschwindigkeiten sollen in einem eigenen Fenster dargestellt werden (z.B. Dashboard)
- Das Fahrzeug sendet eine DENM-Botschaft eines „Emergency Vehicle“ und dann hält an.

Student B:

- Definition einer speziellen Route im „Szenario Manager“ (z.B. in Chemnitz oder Mittweida) (Andere Route als Student A)
- Das Fahrzeug soll auf der Route unterschiedliche Geschwindigkeiten haben
- Diese CAM-Botschaften des Student A beinhalten die Positionen von Student A à das Fahrzeug von Student A soll damit zusätzlich zum eigenen Fahrzeug im Map-Window angezeigt werden
- Die aktuellen Positionen von Student A und B sowie die aktuellen Geschwindigkeiten sollen in einem eigenen Fenster dargestellt werden (z.B. Dashboard)
- Die DENM-Botschaft eines „Emergency Vehicle“ werden empfangen und ausgewertet und Innerhalb von 100 Meter wird eine Warnung angezeigt. Das Fahrzeug des Studenten B hält vor das Fahrzeug des Studenten A an.

Hier ist die Skizze der Versuch2. (Siehe Abb. 36)

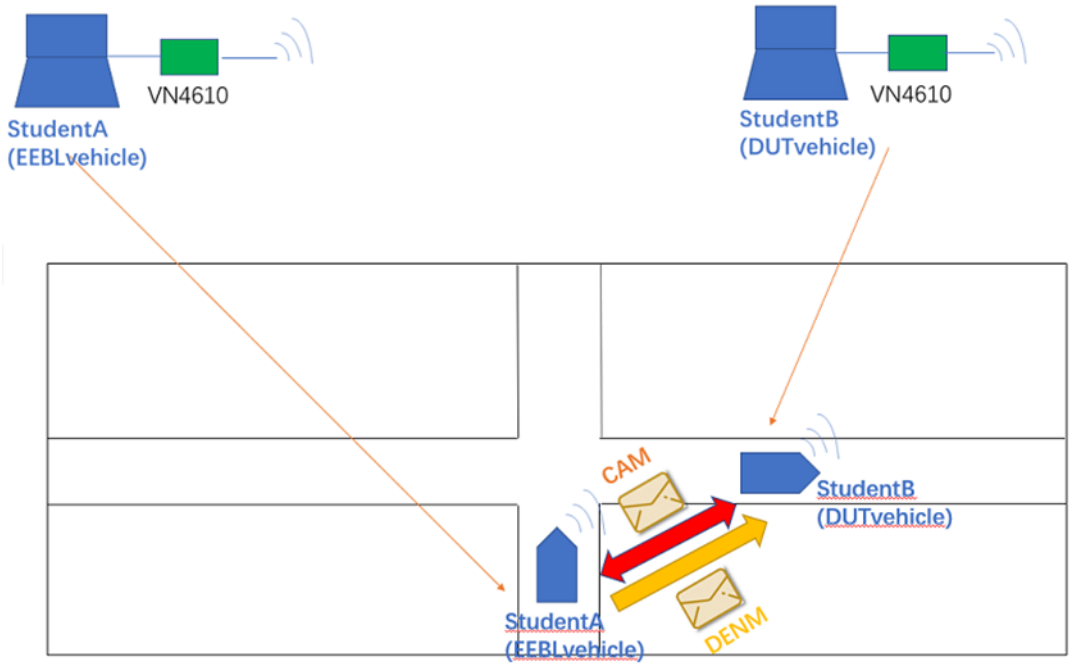


Abbildung 36: Skizze der Versuch2

3.2.2 Projekt erstellen

Zunächst wird eine neue Car2x-Konfiguration erstellt.

Dann wird „EU_ApplMsg.xml“ im Simulationsaufbau eine Car2x-Datenbasis hinzugefügt.

1 Knote wird angelegt (abhängig von Student A oder B). Die Datenbasis sollte Knote beinhalten, welche der Station im Szenarien Editor entsprechen.

- ✓ EEBLvehicle: Das zu testende Fahrzeug der StudentA.
- ✓ DUTvehicle: Das zu testende Fahrzeug der StudentB.

Konfigurieren die Netzwerknote und Tx Nachrichten. Um die Knoten auch im Simulationsaufbau anzuzeigen, werden die Node Sychro-nization (Knoten Sychronisierung) verwendet. Wählen hierzu Node Sychronization über das Kontextmenü der hinzugefügten Datenbasis aus.Die Knote wird anschließend in den Simulationsaufbau eingefügt.

Die Knoten wird über den „Car2x Network Explorer“ angelegt.

Um die Knoten auch im Simulationsaufbau anzuzeigen, werden die Node Sychro-nization (Knoten Sychronisierung) verwendet. Wählen hierzu Node Sychronization über das Kontextmenü der hinzugefügten Datenbasis aus. (Siehe Abb. 37,38)

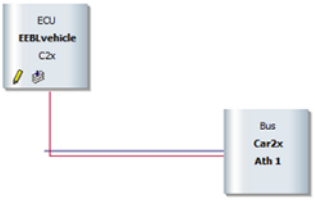


Abbildung 37: Knoten des EEBLvehicles auf PC1

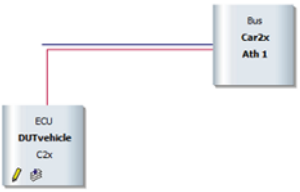


Abbildung 38: Knoten des DUTvehicles auf PC2

3.2.3 Szenario erstellen

Der Car2x Scenario Editor über die Menüleiste in CANoe wird geöffnet, um Scenario zu erstellen.

Im Screenshot rechts wurde eine Route mit 1 Station im Szenarien Editor erstellt. Die Namen der Station müssen mit den Namen der Knoten in der Datenbasis übereinstimmen. Die Farbe der Route und Station im Eigenschaftenfenster kann man selbst bestimmen, um klar zu sehen. Geben der Station unterschiedliche ID. Erstellen ein Verkehrsszenario, indem eine Route und Station (Fahrzeuge) anlegen. Geben der Station unterschiedliche ID.

Beispiel einer Route für EEBL-Fahrzeug (StudentA):

Route3 beginnt auf der Gießerstraße auf der Chemnitzkarte. Fahren etwa 117 Meter geradeaus und biegen rechts ab, um die Markusstraße zu erreichen. Gehen 97 Meter die Markusstraße entlang und biegen dann rechts ab, um Ludwig-Kirsch-Straße zu betreten. Gehen dann 157 Meter die Str. Nationen entlang und biegen auch rechts in die Richard-Tauber-Straße ab. Schließlich kehrt man zum Ursprung zurück. (Siehe Abb. 39)

Abbildung 39: Route3 bestimmen auf PC2

Beispiel einer Route für DUT-Fahrzeug (StudentB):

Route4 beginnt auf der Uhlandstraße auf der Chemnitzkarte. Fahren etwa 103 Meter geradeaus und biegen rechts ab, um die Georgstraße zu erreichen. Gehen 169 Meter die Georgstraße entlang und biegen dann rechts ab, um Str. Nationen zu betreten. Gehen dann 157 Meter die Str. Nationen entlang und biegen auch rechts in die Richard-Tauber-Straße ab. Schließlich kehrt man zum Ursprung zurück. (Siehe Abb. 40)

Abbildung 40: Route4 bestimmen auf PC1

Für die Station werden in der Zeitleiste des Car2x Szenarien Editors Attribute für die Geschwindigkeiten der Fahrzeuge angelegt.

In der 6. Sekunde liegt bei EEBLvehicle ein Notfall vor (Notfallbremsung). Rechtsklicken auf EEBLvehicle und Event wird erstellt, um DENM-Botschaft zu senden.
(Siehe Abb. 41)

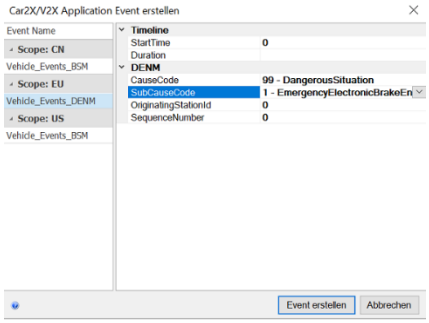


Abbildung 41: CauseCode und SubCausecode

Nach Erstellen des Szenarios speichern dieses im Dateisystem. Es wird empfohlen die Datei relativ zur CANoe Konfiguration bzw. in Ordnern unterhalb dieser zu speichern.

Das erstellte Szenario wird über den „Szenarien Manager“ in CANoe zur Konfiguration hinzugefügt. Der Szenarien Manager wird über die Toolbar in CANoe unter dem Tab Umgebung (Environment) aufgerufen.

In diesem Fenster klicken das Ordner-Symbol in der Menüleiste an und selektieren dann die gespeicherte Szenario-Datei.

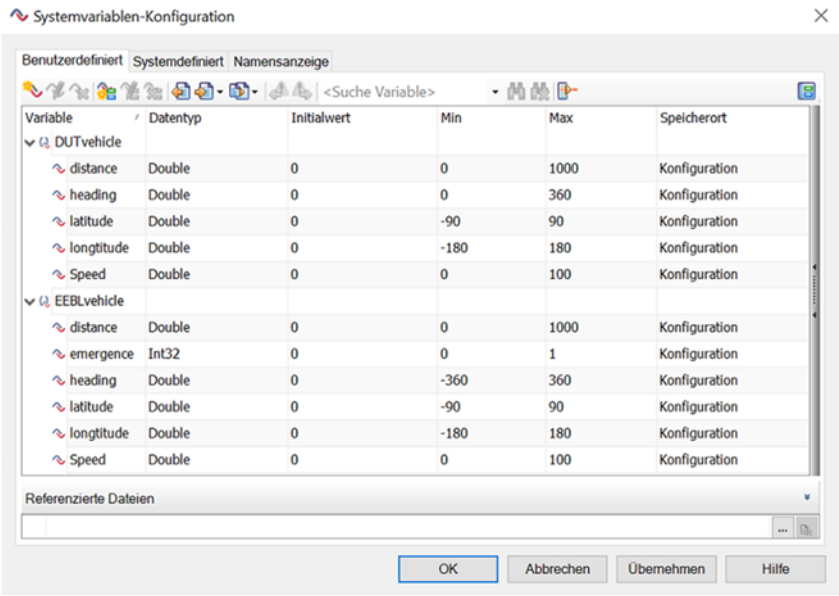
Entsprechen die Namen der Station des Szenarios denen der Knoten in der Datenbank, werden diese im Szenarien Manager mit einem orangenen Icon angezeigt.

3.2.4 Botschaft

Systemvariablen werden häufig von CANoe-Komponenten verwendet. Meistens werden automatisch generiert und können nicht bearbeitet werden. Sie gehören zu einem definierten Namespace [16]. (Siehe Abb. 42) Tabelle 5 zeigt die Definition der Variable und die Definitionen der anderen Variablen sind wie in Versuch 1.

EEBLvehicle::emergence	Es ist eine Flagge, wenn das Fahrzeug in Notfall ist.

Tabelle 5: Die Definition der Systemvariablen in versuch 2



Variable	Datentyp	Initialwert	Min	Max	Speicherort
DUTvehicle					
distance	Double	0	0	1000	Konfiguration
heading	Double	0	0	360	Konfiguration
latitude	Double	0	-90	90	Konfiguration
longtitude	Double	0	-180	180	Konfiguration
Speed	Double	0	0	100	Konfiguration
EEBLvehicle					
distance	Double	0	0	1000	Konfiguration
emergence	Int32	0	0	1	Konfiguration
heading	Double	0	-360	360	Konfiguration
latitude	Double	0	-90	90	Konfiguration
longtitude	Double	0	-180	180	Konfiguration
Speed	Double	0	0	100	Konfiguration

Abbildung 42: Systemvariablen

Nachdem man Systemvariable definiert hat, kann man Botschaft einrichten, um Position, Geschwindigkeiten und Abstand zu senden. (Siehe Abb. 43,44)

EEBLbotschaft

Befehl	Objekt	Operator	Operand	Abst...
Set	...ehicle::Speed	=	100	0
Set	...hicle::latitude	=	50.836457	0
Set	...le::longtitude	=	12.935328	0
Set	...icle::heading	=	16.66	0
Set	...icle::distance	=	206.61	0
Set	emergence	=	0	0
Wait	1000	ms		0
Set	...ehicle::Speed	=	100	0
Set	...hicle::latitude	=	50.836699	0
Set	...le::longtitude	=	12.935443	0

Sequenz ausführbar | Gestoppt

Abbildung 43: Befehl, Objekt, Operator, Operand, Abstand in der 0. Sekunde von EEBLbotschaft

Befehl	Objekt	Operator	Operand	Abst...
Set	...ehicle::Speed	=	100	0
Set	...hicle::latitude	=	50.8382	0
Set	...le::longtitude	=	12.936282	0
Set	...icle::heading	=	108.43	0
Set	...icle::distance	=	206.61	0
Wait	1000	ms		0
Set	...ehicle::Speed	=	100	0
Set	...hicle::latitude	=	50.838121	0
Set	...le::longtitude	=	12.936662	0
Set	...icle::heading	=	108.43	0
Set	...icle::distance	=	180.61	0
Wait	1000	ms		0

Abbildung 44: Befehl, Objekt, Operator, Operand, Abstand in der 0. Sekunde von DUTbotschaft

Um Abstand zu berechnen, muss man eine Regel verwenden.

MLatA bedeutet EEBLvehicle:: latitude, MLatB bedeutet DUTvehicle::latitude

MLonA bedeutet EEBLvehicle::longtitude, MLonB bedeutet DUTvehicle::longtitude

Gleiche Methode wie Versuch 1 kann man den Abstand zwischen EEBLvehicle und DUTvehicle berechnen.

Es dauert 11 Sekunden, bis EEBLvehicle und DUTvehicle eine Runde absolviert haben.

3.2.5 Zertifikate

Im globalen Dialog Optionen werden zentrale Einstellungen für die geladene Konfiguration und für das Programm vornehmen. Im Dialog Zertifikate generieren kann man die Eigenschaften des neuen Zertifikats festlegen. (Siehe Abb. 45) Die Eigenschaft der Zertifikate wie Versuch1 wird erklärt.

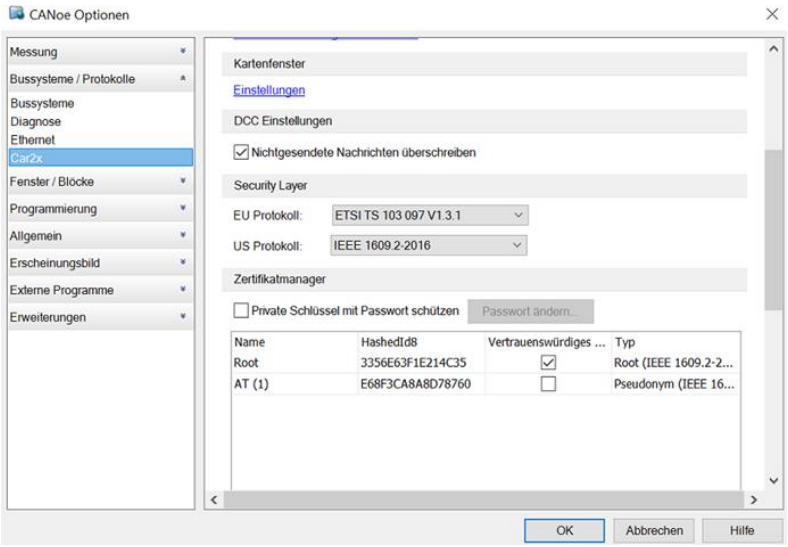


Abbildung 45: Zertifikate auf PC1 und PC2

3.2.6 Panel

Die Positionen, Geschwindigkeit, Abstand werden im Versuch1 durch Panel Design dargestellt. Gleiche Schritte werden gemacht, um neue Dashboards zu erstellen. (Siehe Abb. 46,47)

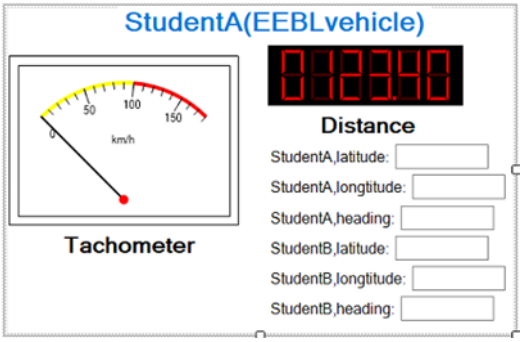


Abbildung 46: Dashboard des EEBLvehicles auf PC1

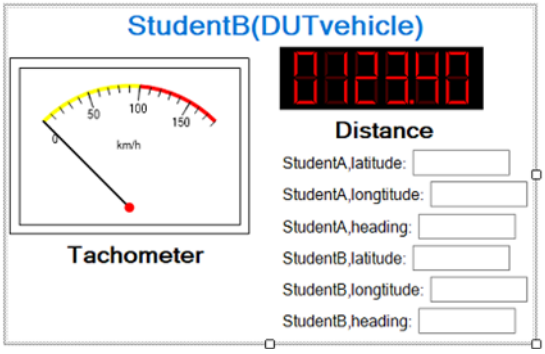


Abbildung 47: Dashboard des DUTvehicles auf PC2

Außerdem muss neu Dashboard designiert werden, um Alarm zu geben. Wenn DUT-vehicle sich dem EEBLvehicle in der dringenden Situation innerhalb 100 Meter begegnet, wird Alarm im Dashboard des DUTvehicles gegeben. (Siehe Abb. 48)

Abbildung 48: Dashboard des EEBLvehicles auf PC1

3.3 Versuch3-Kommunikation mit RSU

3.3.1 Ziel des Versuchs 3

Ampel ordnet für Verkehrsteilnehmer ein bestimmtes Verhalten an, indem sie gesteuerte Signale abgeben. In diesem Versuch soll die Kommunikation zwischen Fahrzeug und einer Ampelanlage simuliert werden.

TrafficLight:

- Die Zeit der roten Ampel, roten gelben Ampel, gelben Ampel, Grünen Ampel einrichten.
- Sendet SPAT-Nachricht.

- Die aktuelle Ampel und die Zeit über nächste kommende Ampel sollen in einem eigenen Fenster dargestellt werden (z.B. Dashboard)

StudentA:

- Empfängt SPAT-Nachricht.
- Wartet auf der roten Ampel und geht während der grünen Ampel.
- Der aktuelle Abstand zwischen Student A, die aktuelle Ampel und die Zeit über nächste kommende Ampel sollen in einem eigenen Fenster dargestellt werden (z.B. Dashboard)

Hier ist die Skizze des Versuchs 3. (Siehe Abb. 49)

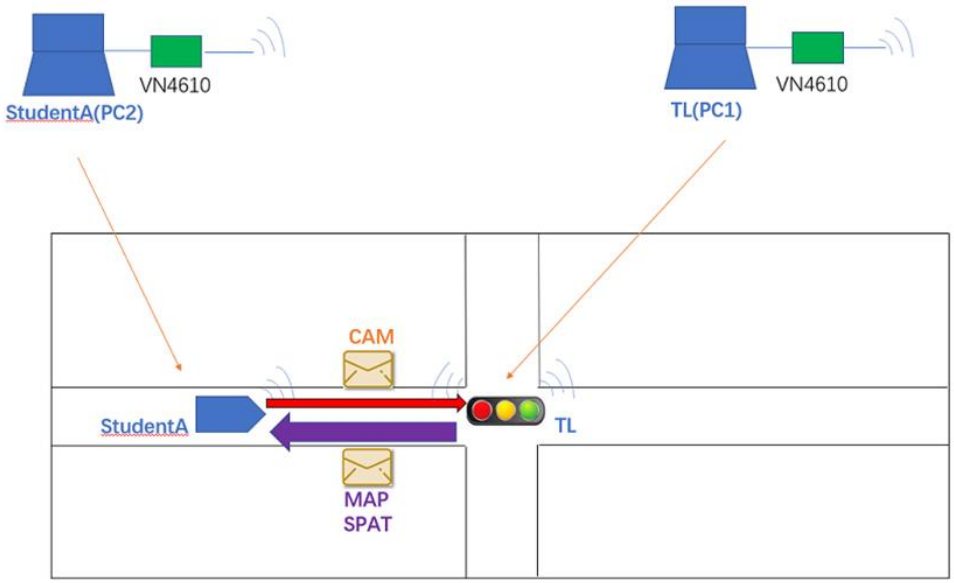


Abbildung 49: Skizze der Versuch3

3.3.2 Projekt erstellen

Zunächst wird eine neue Car2x-Konfiguration erstellt.

Dann wird „EU_ApplMsg.xml“ im Simulationsaufbau eine Car2x-Datenbasis hinzugefügt.

1 Knoten wird angelegt (abhängig von Student A oder TL). Die Datenbasis sollte Knoten beinhalten, welche der Station im Szenarien Editor entsprechen.

- ✓ TL: TrafficLight.

✓ StudentA: Das zu testende Fahrzeug der StudentA.

Konfigurieren die Netzwerknote und Tx Nachrichten.

Um die Knoten auch im Simulationsaufbau anzuzeigen, werden die Node Sychronization (Knoten Sychronisierung) verwendet. Wählen hierzu Node Sychronization über das Kontextmenü der hinzugefügten Datenbasis aus. Der Knote wird anschließend in den Simulationsaufbau eingefügt.

Die Knoten wird über den „Car2x Network Explorer“ angelegt. (Siehe Abb. 50,51)

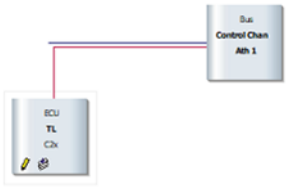


Abbildung 50: Knote auf PC1

Abbildung 51: Knote auf PC2

3.3.3 Szenario erstellen

Der Car2x Scenario Editor über die Menüleiste in CANoe wird geöffnet, um Scenario zu erstellen.

Im Screenshot rechts wurde eine Route mit 1 Station im Szenarien Editor erstellt. Die Namen der Station müssen mit den Namen der Knoten in der Datenbasis übereinstimmen. Die Farbe der Route und Station im Eigenschaftenfenster werden selbst bestimmt, um klar zu sehen. Geben der Station unterschiedliche ID. Erstellen ein Verkehrsszenario, indem eine Route und Station (Fahrzeuge) anlegen.

Geben der Station unterschiedliche ID.

Route1 beginnt auf der Limbachstraße. Fahren etwa 285 Meter geradeaus und betreten die Hartmannstraße. (Siehe Abb. 52)

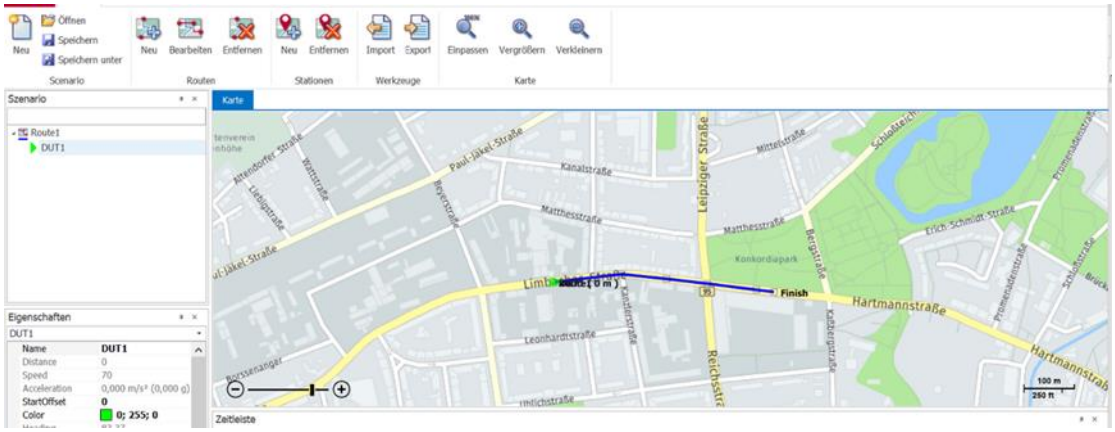


Abbildung 52: Route bestimmen

Für die Station werden in der Zeitleiste des Car2x Szenarien Editors Attribute für die Geschwindigkeiten der Fahrzeuge angelegt.

Das erstellte Szenario wird über den „Szenarien Manager“ in CANoe zur Konfiguration hinzugefügt. Der Szenarien Manager wird über die Toolbar in CANoe unter dem Tab Umgebung (Environment) aufgerufen.

In diesem Fenster klicken das Ordner-Symbol in der Menüleiste an und selektieren dann die gespeicherte Szenario-Datei.

Entsprechen die Namen der Station des Szenarios denen der Knoten in der Datenbank, werden diese im Szenarien Manager mit einem orangenen Icon angezeigt. (Siehe Abb. 53)

Abbildung 53: Car2x Scenario Manager

3.3.4 Programmierung

Systemvariablen werden häufig von CANoe-Komponenten verwendet. Meistens werden automatisch generiert und können nicht bearbeitet werden. Sie gehören zu einem definierten Namespace.[16]

Beispiele Definition:

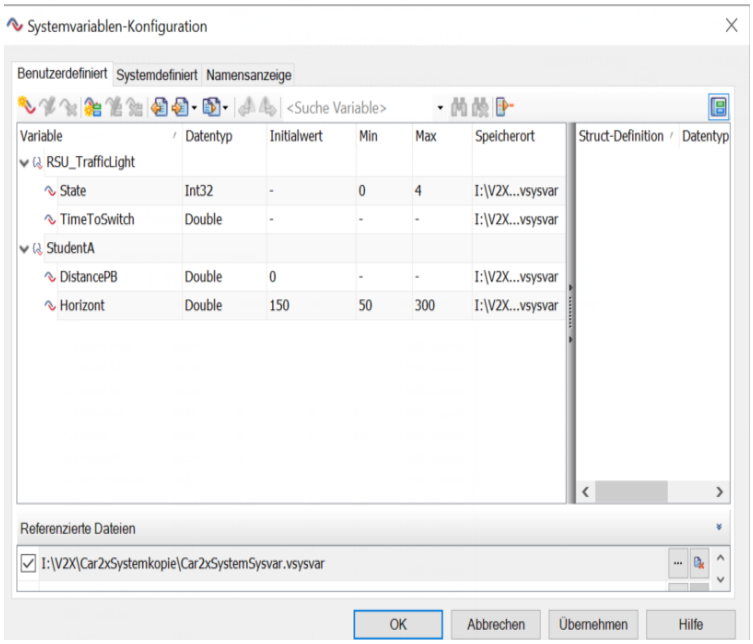


Abbildung 54: Systemvariablen-Konfiguration

Tabelle 5 zeigt die Definition der Variablen.

RSU_TrafficLight::State	Es steuert 4 Zustände von Ampel: rot, rotgelb, gelb, grün.
RSU_TrafficLight::TimeToSwitch	Es zeigt, wie lange es bis zur nächsten Ampel dauern wird?
StudentA::DistancePB	Es bedeutet den Abstand zwischen StudentA und Ampel.
StudentA::Horizont	Es zeigt an, dass das Fahrzeug die Grenzen des Signals der Ampel akzeptieren kann.

Tabelle 6: Die Definition der Systemvariablen in versuch 3

Nachdem Systemvariable (Siehe Abb. 54) definiert werden, wird Botschaft (Siehe Abb. 55) einrichten, um den Abstand zu senden.

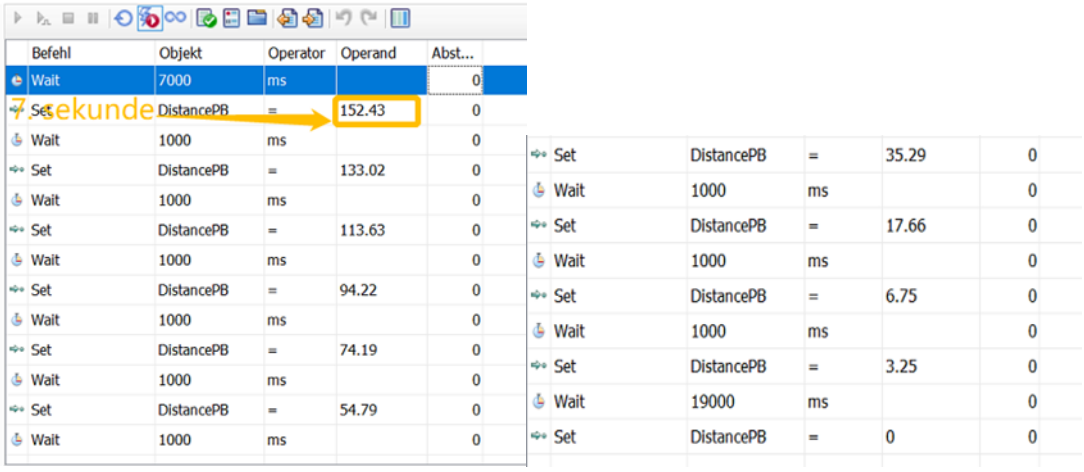


Abbildung 55: Botschaft

Drei Schriftgutbehälter werden über Programmierung aus Car2x System kopiert. Für TL ist RSU_TrafficLight.can.

Diese Programmierung wurde kompiliert. Es scheint einen Fehler auf Writer-Fenster. „DUT“ wird korrigiert. (Siehe Abb. 56)

Diese Programmierungen werden das Fahrzeug von der Ampel empfangen. Wenn das Fahrzeug innerhalb des Signalbereichs der Ampel fährt, erhält es Botschaft.

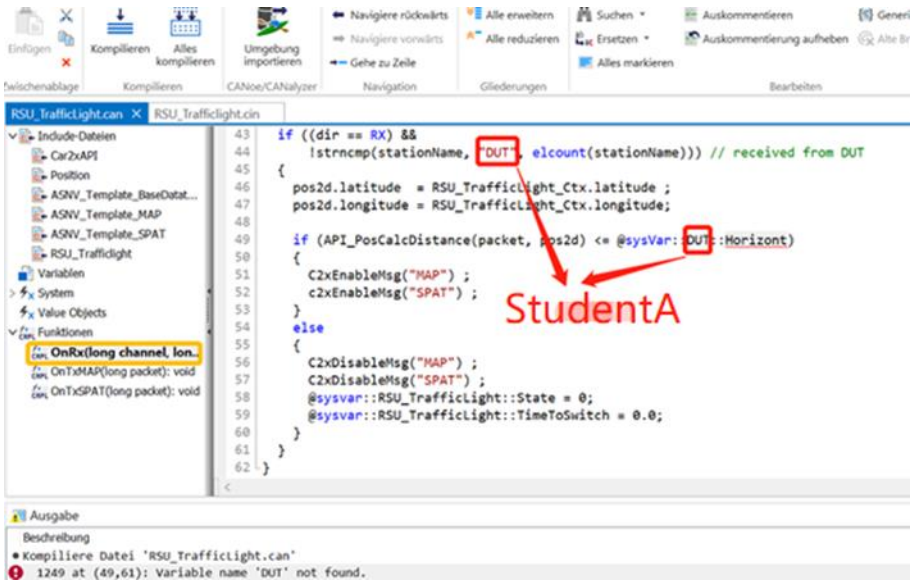


Abbildung 56: Programmierung1 aus RSU_TrafficLight.can

Danach wird der Code in der 108. Zeile entfernt, weil signalGroup_2 für andere TrafficLight. (Siehe Abb. 57)

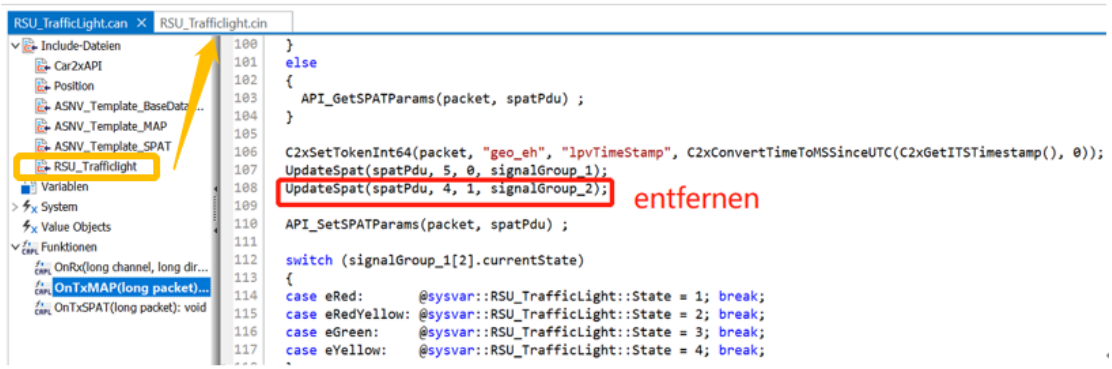


Abbildung 57: Programmierung2 aus RSU_TrafficLight.can

Doppelklicken „RSU_TrafficLight“ und Untergeordnete Datei wird geöffnet.

Die Zeile von 82 bis 112, von 331 bis 449 und von 515 bis 535 werden auch entfernt, was für andere Intersection. Der Code geht es um `signalGroup_2[4]`, `mapPdu.map.intersections.arrayValue[1]`,`spatPdu.spat.intersetions.arrayValue[1]`

Sehen zuerst das Beispiel. Der grüne Punkt im Abbildung 57 ist das Gerät, das Botschaft sendet und empfängt. Die Straßenkreuzung (Intersection ID1) besteht aus 10 Lane. (Siehe Abb. 58)

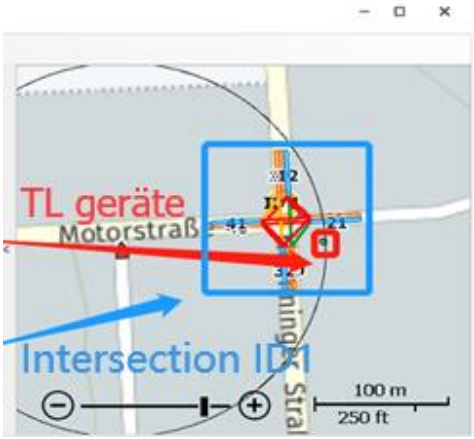


Abbildung 58: Map Window aus dem Beispiel „Car2xSystem“ (1)

Wenn TrafficLight sich von Stuttgart zu Chemnitz geändert wird, muss die Position des TLs und die Position des Intersection ID bestimmt werden. In der 17. Zeile zeigt die Latitude, Longtitude, Elevation des TLs. (Siehe Abb. 59)

Abbildung 59: Map Window aus dem Beispiel „Car2xSystem“ (2)

Auf Chemnitzkarte wird ein geeigneter Punkt als Standort des TLs gewählt. Das Zentrum der Straßenkreuzung ist als ID1. Genaue Position wird durch Car2x Scenario Editor gewusst. (Siehe Abb. 60)

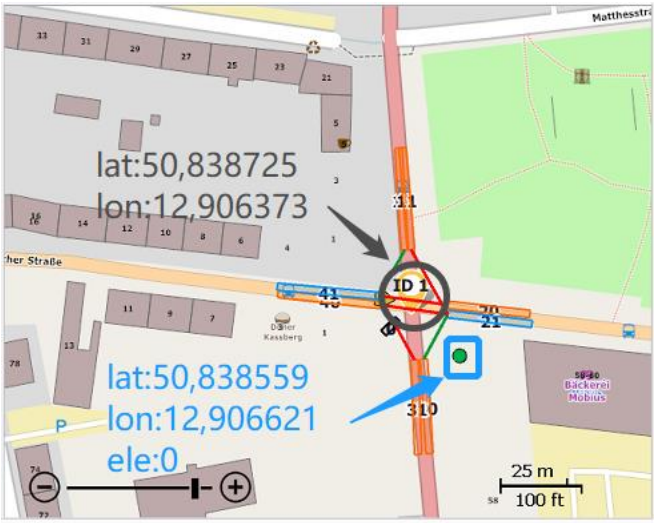


Abbildung 60: Neue TL und Intersection in Chemnitz

Dann verändert der Code sich in RSU_TrafficLight.cin. Hier sind die Programmierungen des TLs. Eine neue Position des TLs wird in Chemnitz geschrieben. (Siehe Abb. 61)

```
17  struct RSU_TrafficLight_T RSU_TrafficLight_Ctx =
18  {
19    /* Latitude */ 50.838559,
20    /* Longitude */ 12.906621,
21    /* Elevation */ 0,
22    /* Heading */ 0,
23    /* Speed */ 0
24  };
```

Abbildung 61: Neue Position des TLs in Chemnitz

Hier sind Code über Intersection. Eine neue Position des refpoints wird in Chemnitz geschrieben. (Siehe Abb. 62)

```
123  mapPdu.map.intersections.arrayValue[0].id.id = 1;
124  mapPdu.map.intersections.arrayValue[0].revision = 0;
125  mapPdu.map.intersections.arrayValue[0].refPoint.lat = 508387250.0;
126  mapPdu.map.intersections.arrayValue[0].refPoint.lon = 129063730.0;
127  mapPdu.map.intersections.arrayValue[0].laneWidth.isValidFlag = 1;
128  mapPdu.map.intersections.arrayValue[0].laneWidth.value = 350;
129  mapPdu.map.intersections.arrayValue[0].speedLimits.isValidFlag = 1;
130  mapPdu.map.intersections.arrayValue[0].speedLimits.length = 1;
131  mapPdu.map.intersections.arrayValue[0].speedLimits.arrayValue[0].type = 5; // vehicleMaxSpeed
132  mapPdu.map.intersections.arrayValue[0].speedLimits.arrayValue[0].speed = 695;
133  mapPdu.map.intersections.arrayValue[0].laneSet.length = 10;
134  mapPdu.map.intersections.arrayValue[0].laneSet.arrayValue[0].laneID = 10;
135  mapPdu.map.intersections.arrayValue[0].laneSet.arrayValue[0].laneAttributes.directionalUse.stringLength = 2;
```

Abbildung 62: Neue Position des refpoints in Chemnitz

Wenn nur die Position des TLs und Intersection ID1s, nicht die Position der Lane verändert wird, wird es so wie Abb.63 scheinen. Nächste Schritt wird die Lane umgestellt. (Siehe Abb. 63) Denn die Straße in Chemnitz ist enger als die in Stuttgart. Deshalb verändert die Anzahl der Lane von 10 bis 8.

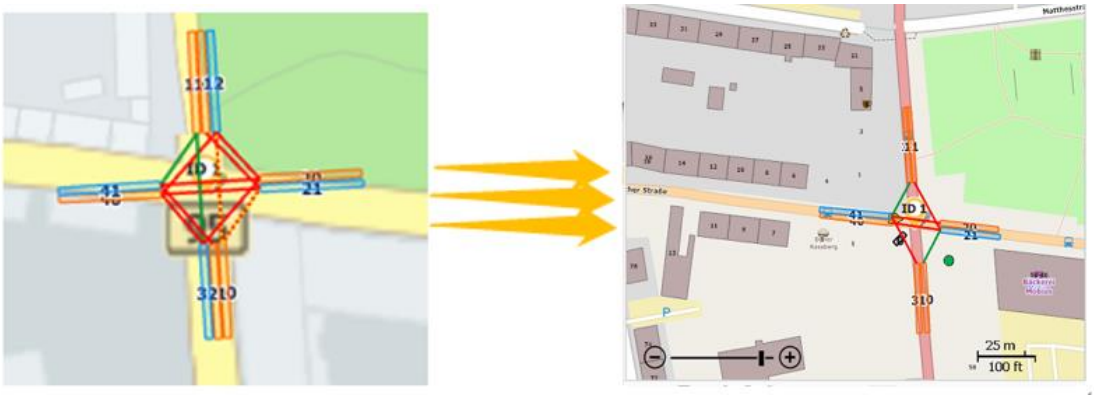


Abbildung 63: Von originaler Lane bis gezielter Lane

Lane sind auf ID1 zentriert. Die horizontale Linie ist die X-Achse und die vertikale Linie ist die y-Achse. Die Programmierungen der Position der Lane ID10 werden gefunden. Mit x.0, y.0, x.1, y.1 wird die Lane genaue Position mit schwarzen Schriften auf Abb.

58 bestimmt. Nach dem Versuch die entsprechende Koordinate mit roten Schriften auf Abb. 98 gefunden werden. (Siehe Abb. 64)

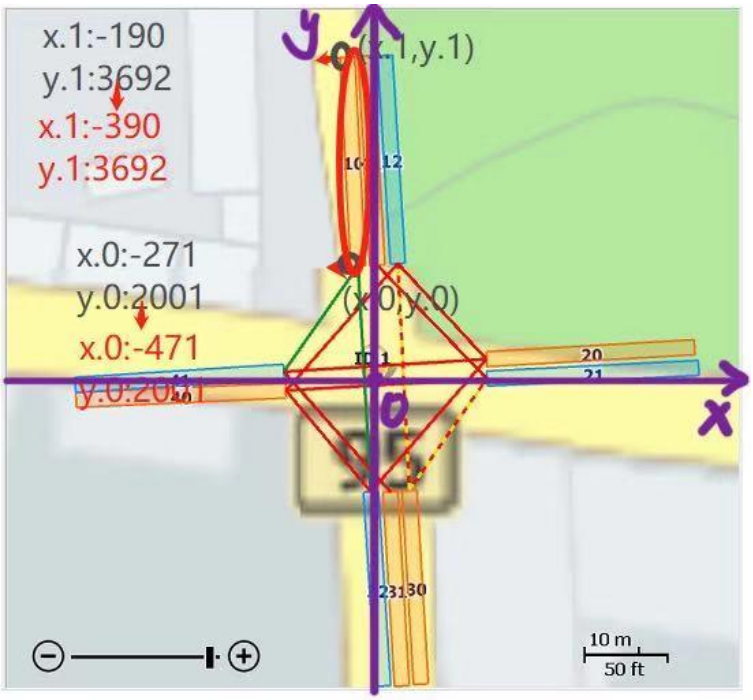


Abbildung 64: Position der Lane

Die Programmierungen der Lane ID10 werden sich geändert. (Siehe Abb. 65)

```
lue[0].laneSet.arrayValue[0].laneID = 10;
lue[0].laneSet.arrayValue[0].laneAttributes.directionalUse.stringLength = 2;
.arrayValue[0].laneSet.arrayValue[0].laneAttributes.directionalUse.string, "10", 3);
lue[0].laneSet.arrayValue[0].maneuvers.isValidFlag = 1;
lue[0].laneSet.arrayValue[0].maneuvers.stringLength = 3;
.arrayValue[0].laneSet.arrayValue[0].maneuvers.string, "101", 13);
lue[0].laneSet.arrayValue[0].nodeList.nodes.length = 2;
lue[0].laneSet.arrayValue[0].nodeList.nodes.arrayValue[0].delta.choice = 3;
lue[0].laneSet.arrayValue[0].nodeList.nodes.arrayValue[0].delta.node_XY4.x = -251.6812054999866;
lue[0].laneSet.arrayValue[0].nodeList.nodes.arrayValue[0].delta.node_XY4.y = 2001.713731838673;
lue[0].laneSet.arrayValue[0].nodeList.nodes.arrayValue[1].delta.choice = 3;
lue[0].laneSet.arrayValue[0].nodeList.nodes.arrayValue[1].delta.node_XY4.x = -230.91111737519628
lue[0].laneSet.arrayValue[0].nodeList.nodes.arrayValue[1].delta.node_XY4.y = 3692.049772054485;
```

Abbildung 65: Programmierung der Lane ID10

Gleiche Schritte werden die Position der Lane ID 11,20,21,30,31,40,41 verändern. (Siehe Tabelle 4)

Lane ID 11	arrayValue[0].delta.node_XY4.x = -27.42735283360396;
	arrayValue[0].delta.node_XY4.y = 2012.8343636541263;
	arrayValue[1].delta.choice = 3;
	arrayValue[1].delta.node_XY4.x = -312.9393232226688;
	arrayValue[1].delta.node_XY4.y = 3703.1704039489555;

Lane ID 20	arrayValue[0].delta.node_XY4.x = 1300.5667324330395;
	arrayValue[0].delta.node_XY4.y = 11.86021896369107;
	arrayValue[1].delta.choice = 3;
	arrayValue[1].delta.node_XY4.x = 2986.053112379961;
	arrayValue[1].delta.node_XY4.y = -333.2920050467307;

Lane ID 21	arrayValue[0].delta.node_XY4.x = 1311.9094677111825;
	arrayValue[0].delta.node_XY4.y = -333.120631815453306;
	arrayValue[1].delta.choice = 3;
	arrayValue[1].delta.node_XY4.x = 3052.1377299354212;
	arrayValue[1].delta.node_XY4.y = -355.05074133680725;

Lane ID 30	arrayValue[0].delta.node_XY4.x = 438.8179696680065;
	arrayValue[0].delta.node_XY4.y = -1957.2312044978432;
	arrayValue[1].delta.choice = 3;
	arrayValue[1].delta.node_XY4.x = 205.59658794452577;
	arrayValue[1].delta.node_XY4.y = -3436.275239587908;

Lane ID 31	arrayValue[0].delta.node_XY4.x = 101.05214662560223;
	arrayValue[0].delta.node_XY4.y = -1990.5931000232197;
	arrayValue[1].delta.choice = 3;
	arrayValue[1].delta.node_XY4.x = 135.59658794452577;
	arrayValue[1].delta.node_XY4.y = -3436.275239587908;

Lane ID 40	arrayValue[0].delta.node_XY4.x = -1149.317144849043;
	arrayValue[0].delta.node_XY4.y = 0.17137315226057;
	arrayValue[1].delta.choice = 3;
	arrayValue[1].delta.node_XY4.x = -3715.4240535186195;
	arrayValue[1].delta.node_XY4.y = 400.80947770590066;

Lane ID 41	arrayValue[0].delta.node_XY4.x = -1186.030821265845;
	arrayValue[0].delta.node_XY4.y = 333.44758210150704;
	arrayValue[1].delta.choice = 3;
	arrayValue[1].delta.node_XY4.x = -3693.3958476711473;
	arrayValue[1].delta.node_XY4.y = 333.53326867763732;

Tabelle 7: Änderungen der Lane

Nachdem die Positionen der TLs und Intersection ID1 bestimmt wird, wird die Zeit der unterschiedlichen Ampel festgelegt. Unterschiedliche Ampel-Gruppe können bestimmt werden. Es gibt 5 Gruppe. Der Code in Zeile 120 kann in RSU_TrafficLight.can bestimmt werden. Die unterschiedliche Ampel für unterschiedlichen Zeiten wird auch sich durch die Programmierung (Siehe Abb. 66) ändern.

Abbildung 66: Ampel-Gruppe bestimmen

Erstens ist das rotes Light für 27 s, dann ist rot-gelbes Light für 3 s und danach leuchtet grünes Light für 6s. Das gelbes Light leuchtet für 3s. Das rotes Light leuchtet noch für 27s. (Siehe Abb. 66)

```
RSU_Trafficlight.cin ×
43
44  struct SignalGroup signalGroup_1[5] =
45  {
46    {
47      /* id */ 1,
48      /* stateTime */ { 21, 3,  12,  3 },
49      /* currentState */ eRedYellow,
50      /* nextStateChange */ 3,
51      /* permissiveMovement */ 0
52    },
53    {
54      /* id */ 2,
55      /* stateTime */ { 30,  3,  3,  3 },
56      /* currentState */ eRedYellow,
57      /* nextStateChange */ 3,
58      /* permissiveMovement */ 1
59    },
60    {
61      /* id */ 3,
62      /* stateTime second red 27s,redyellow 3s,green 6s,yellow 3s */ {  27,  3,  6,  3 },
63      /* currentState */ eRed,
64      /* nextStateChange first red 27s */ 27,
65      /* permissiveMovement */ 0
66    },
67    {
```

signalGroup_1[2]

Abbildung 67: Unterschiedliche Ampel für unterschiedlicher Zeit

3.3.5 Zertifikate

Im globalen Dialog Optionen werden zentrale Einstellungen für die geladene Kon-figuration und für das Programm vornehmen. (Siehe Abb. 68) Die Eigenschaft der Zertifikate wie Versuch1 wird erklärt.

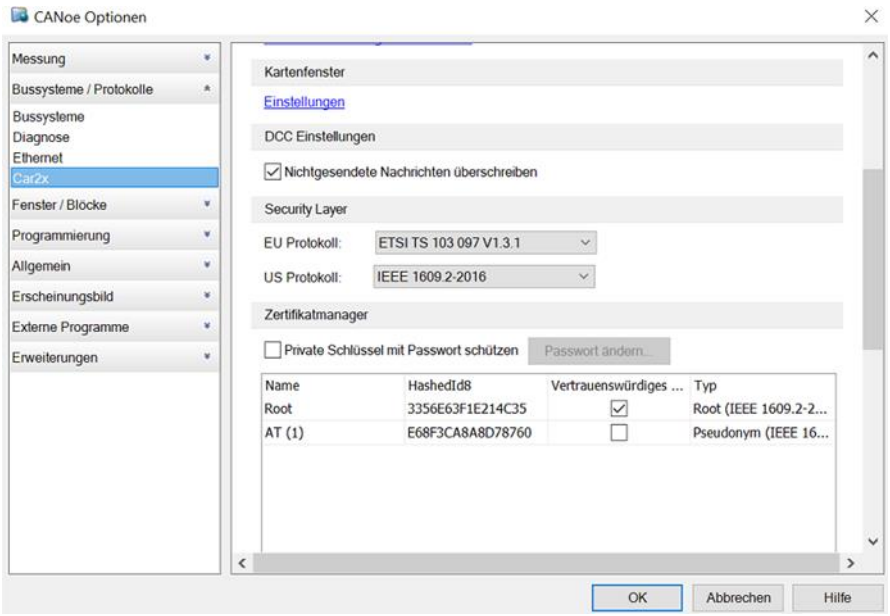


Abbildung 68: Zertifikate

3.3.6 Panel

Panel für das Fahrzeug und TL. (Siehe Abb. 69,70) Es ist zu beachten, dass Traffic Light Switch ist und konfiguriert den Traffic Light im Eigenschaften-Fenster unter General|Display Only mit True. Konfigurieren den Traffic Light im Eigenschaften-Fenster unter Switch Values|State Count als Element mit 5. Weisen per Drag-and-Drop aus dem Symbol Explorer die Systemvariable RSU_TrafficLight::State dem Traffic Light zu.

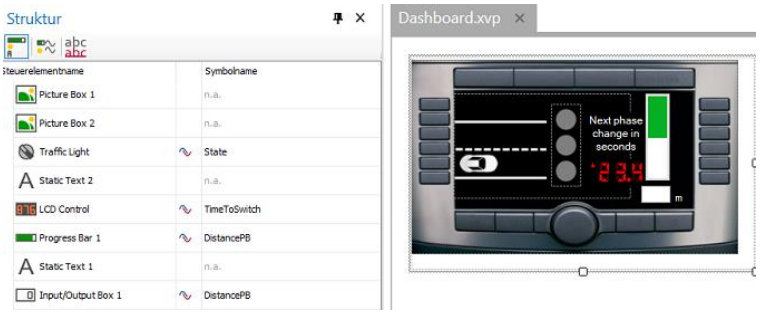


Abbildung 69: Dashboard

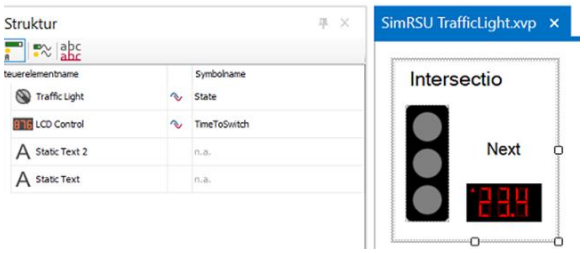


Abbildung 70: SimRSU TrafficLight

4 Evaluierung der Versuche

4.1 Test des Versuchs 1

4.1.1 Simulation auf einem PC

Den Versuch werden zunächst auf einem Computer getestet („Simulierter Bus“-Modus). Danach werden zwei Konfigurationen erstellt (Student A: lec-pc3-01 / Student B: lec-pc3-02) und auf beide Rechner im „Realer Bus“-Modus getestet.

4.1.2 Simulation zwischen zwei Knoten

Um CANoe mit realem Bus zu kommunizieren. Zunächst wird Bus-Hardware eingerichtet. Der Dialog Netzwerk-Hardware-Konfiguration wird geöffnet. (Siehe Abb. 71) Um die gegenseitigen Störungen bzw. Interferenzen zwischen Car2x (DSRC) und ITS-G5A/B (Mautstationen) Funkkommunikation zu vermeiden, werden beim Versenden von Car2x Frames die Koexistenz-Regeln (reduzierte Sendeleistung) nach ETSI TS 102 792 angewendet. Die verfügbare maximale Sendeleistung abhängt aufgrund der regulatorischen Anforderungen vom Ländercode. (10dBm EIRP für die EU oder 19dBm EIRP für die USA). [14]

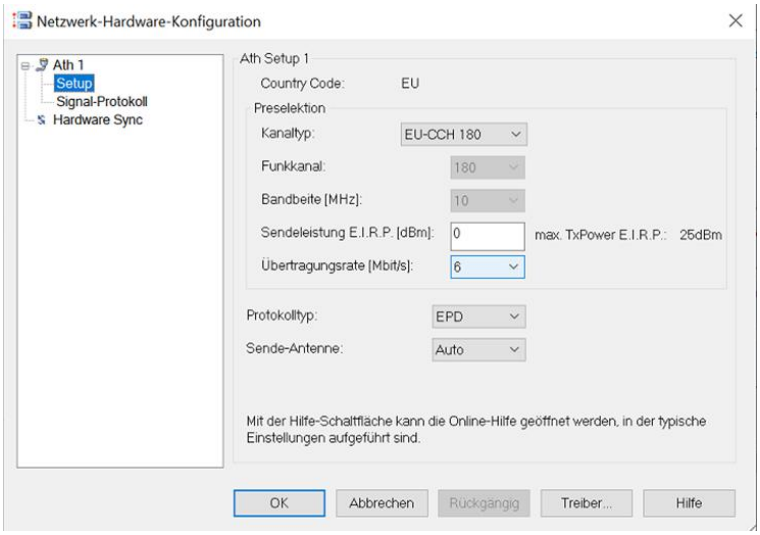


Abbildung 71: Netzwerk-Hardware Konfiguration

Es läuft gut. Aber es gibt die Sendewarnungen. Die Warnung sagt, dass identische Nachrichten verschickt werden. (Siehe Abb. 72)

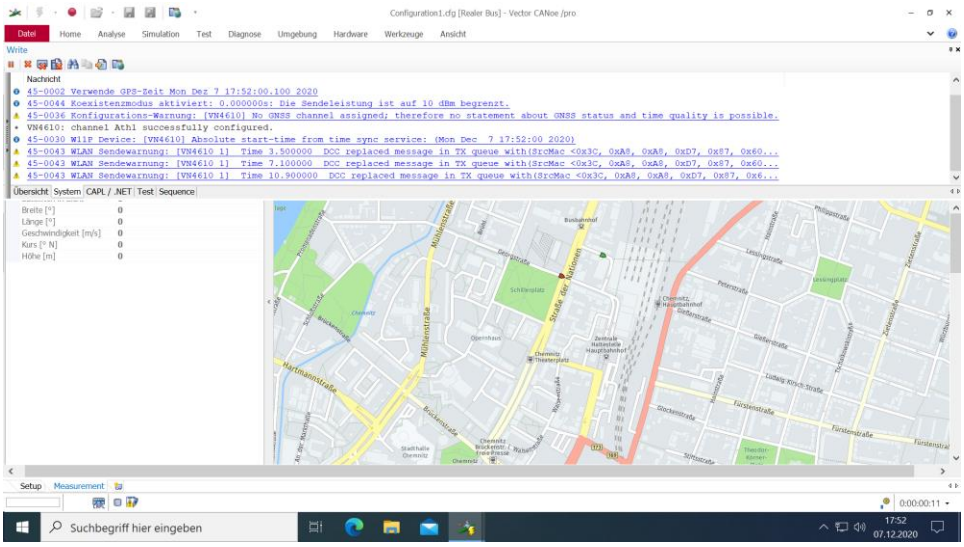


Abbildung 72: Sendewarnungen in Versuch1

Es lässt das Szenario auf beide Computer gleichzeitig laufen. Das ist richtig, aber es muss auf PC1 zum Beispiel Fahrzeug 2 des Studenten B deaktivieren und auf PC2 das Fahrzeug 1 des Studenten A deaktivieren. (Siehe Abb. 73,74).

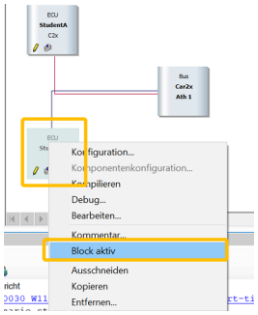


Abbildung 73: Fahrzeug 2 des Studenten B auf PC1 deaktivieren

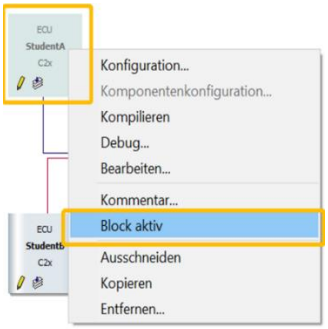


Abbildung 74: Fahrzeug 1 des Studenten A auf PC2 deaktivieren

Dann startet man CANoe und das Problem hat gelöscht werden.

StudentA auf PC 1 sendet seine Position und empfängt die Position von StudentB. StudentB auf PC 2 sendet seine Position und empfängt die Position von StudentA. Die aktuellen Positionen von Student A und B sowie die aktuellen Geschwindigkeiten sollen auf Panel dargestellt werden. (Siehe Abb. 75,76)

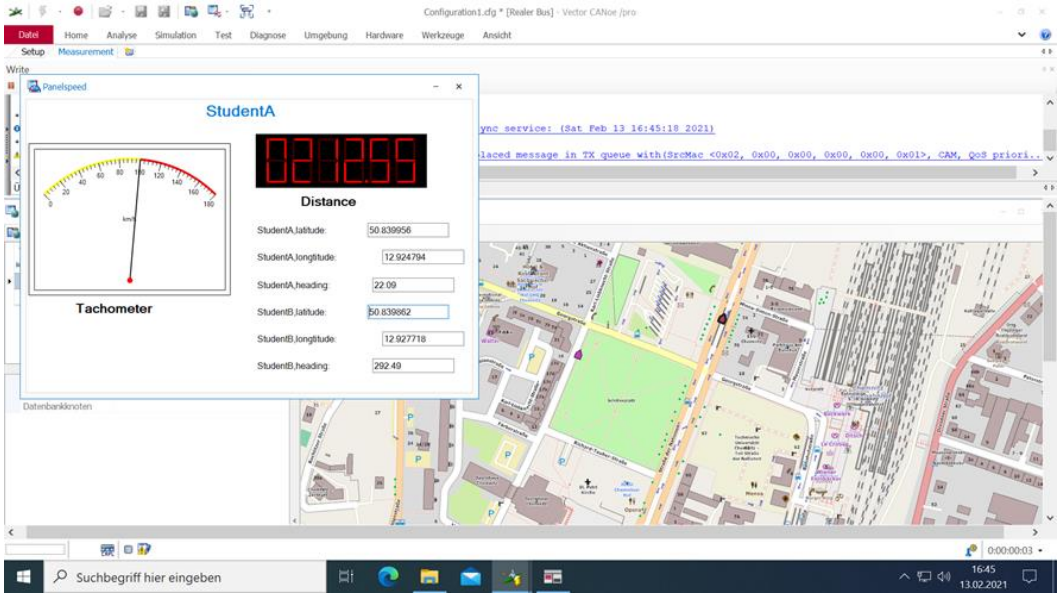


Abbildung 75: StudentA auf PC1

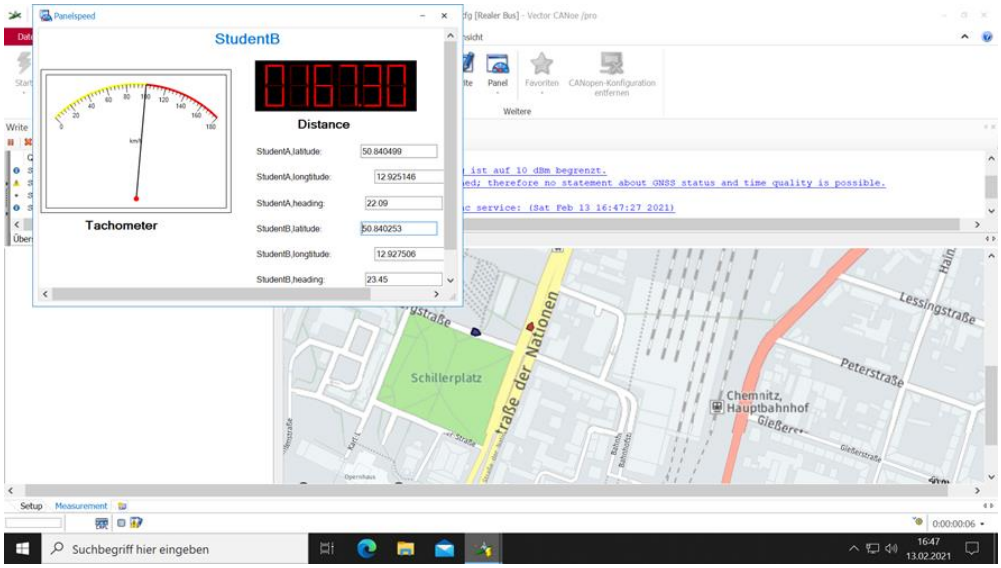


Abbildung 76: StudentB auf PC2

4.2 Test des Versuchs 2

4.2.1 Simulation auf einem PC

Gleiche Schritte wie Versuch1. Zuerst werden beide Computer simulieret. Danach werden zwei Konfigurationen erstellt (Student A: lec-pc3-01 / Student B: lec-pc3-02) und auf zwei Computers im „Realer Bus“-Modus getestet.

4.2.2 Simulation zwischen zwei Knoten

Die aktuellen Positionen von Student A sowie die aktuellen Geschwindigkeiten haben in einem eigenen Panel dargestellt werden. Das Fahrzeug hat eine DENM-Botschaft eines „Emergency Vehicle“ gesendet und dann hat angehalten. (Siehe Abb. 77)

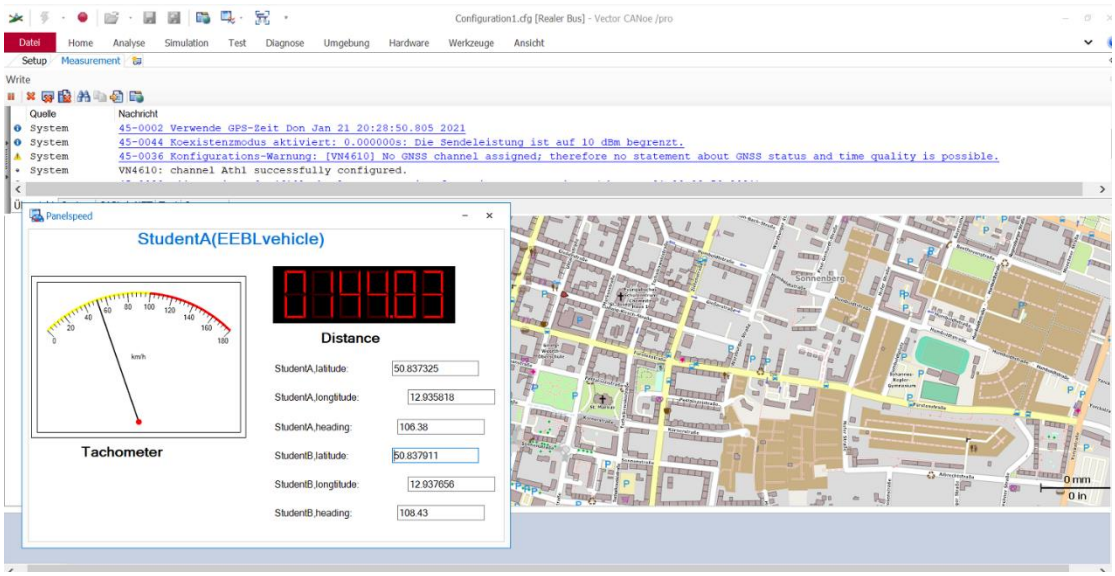


Abbildung 77: EEBLvehicle(StudentA) auf PC1

Die aktuellen Positionen von Student B sowie die aktuellen Geschwindigkeiten haben in einem eigenen Panel dargestellt werden. Die DENM-Botschaft eines „Emergency Vehicle“ werden empfangen und ausgewertet und Innerhalb von 100 Meter wird eine Warnung angezeigt. Das Fahrzeug des Studenten B hat vor das Fahrzeug des Studenten A angehalten. (Siehe Abb. 78)

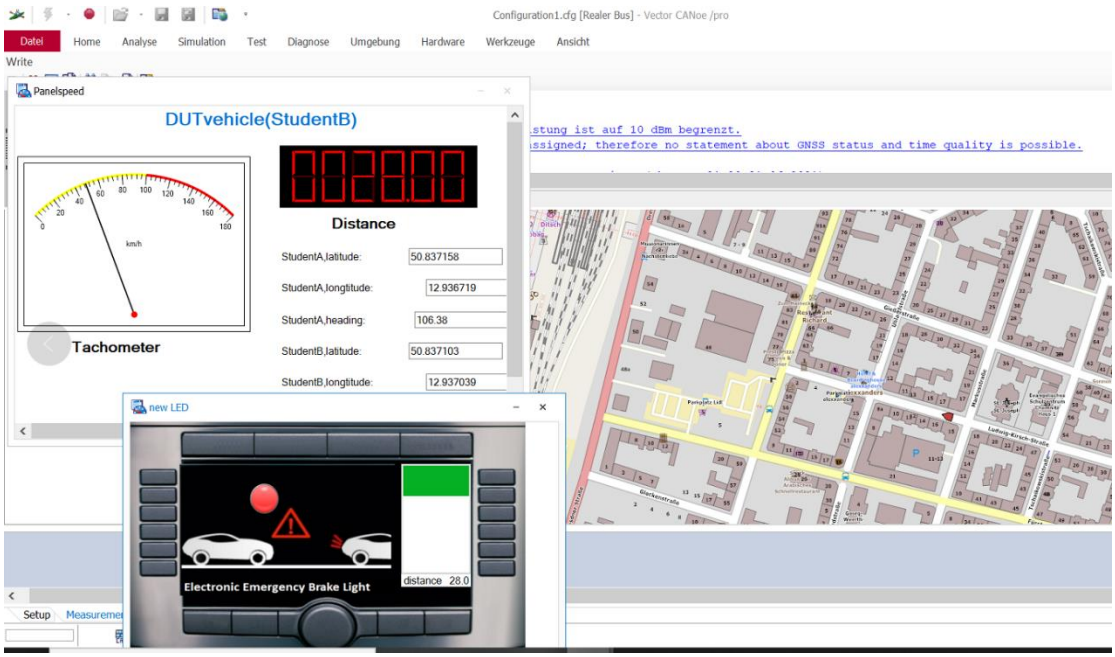


Abbildung 78: DUTvehicle(StduentB) auf PC2

4.3 Test des Versuchs 3

4.3.1 Simulation auf einem PC

Gleiche Schritte wie Versuch1. Zuerst werden beide Computer simulieret. Danach werden zwei Konfigurationen erstellt (DUT1: lec-pc3-01 / TL: lec-pc3-02) und auf zwei Computers im „Realer Bus“-Modus getestet.

In Overview gibt es irrelevante Fenster, Stationen entfernen. Hier ist mein Endergebnis. (Siehe Abb. 79)

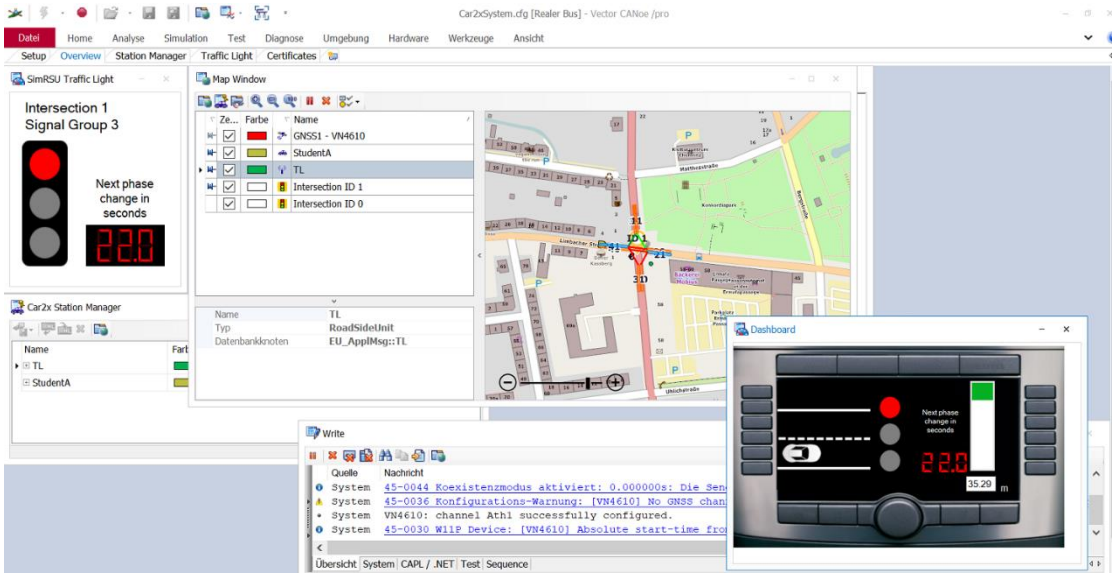


Abbildung 79: Endergebnis

4.3.2 Simulation zwischen zwei Knoten

Wenn alles gut funktioniert, wird man wie Abb. 80,81 sehen. Denn es gibt keine Programmierung von TL auf PC 2. Deshalb zeigt Dashboard auf PC2 kein Signal von Ampel.

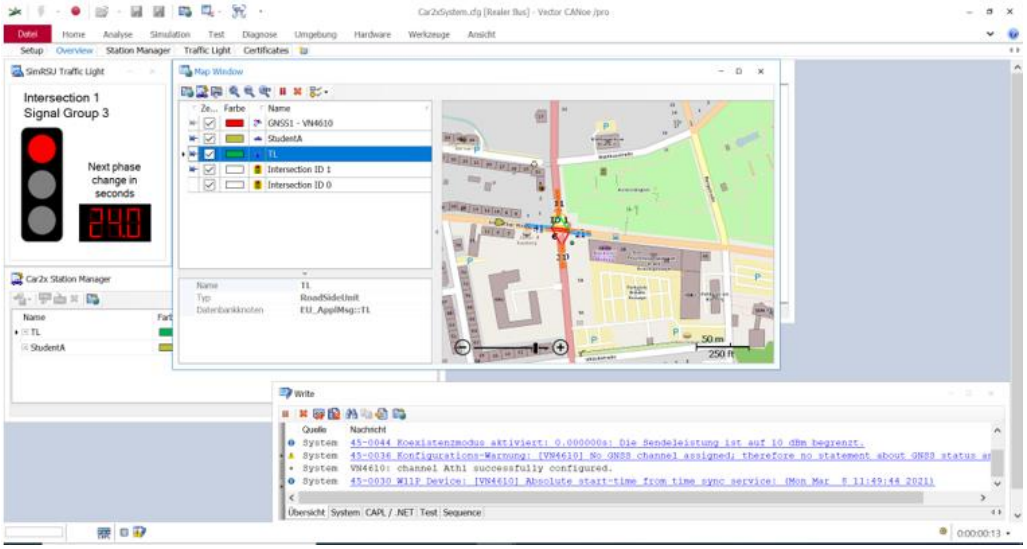


Abbildung 80: TL auf PC1

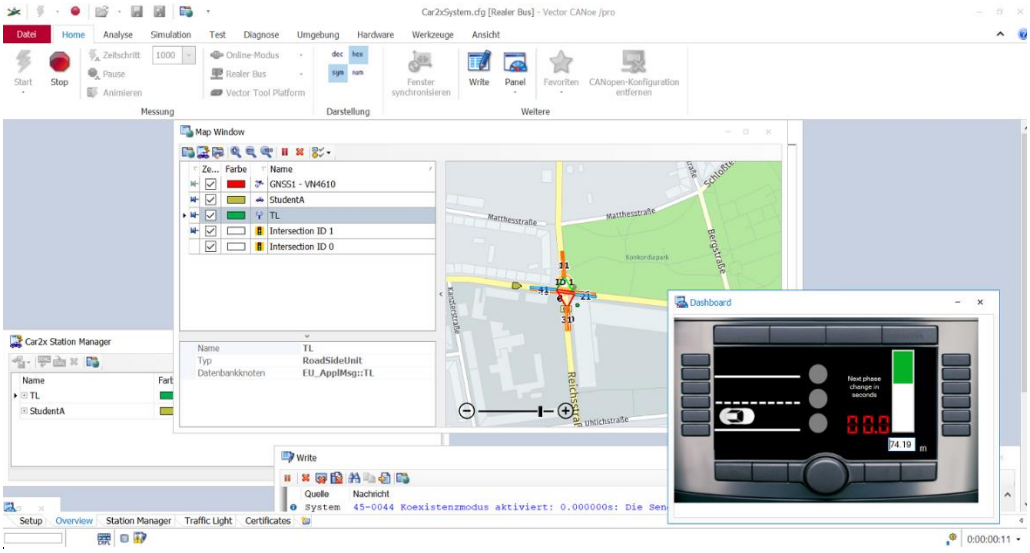


Abbildung 81: StudentA auf PC2

Früh funktionierten das Fahrzeug und Ampel gut. Aber es gab ein Problem: Certifikate not found. Denn keine Knoten wurden erstellt. Der originalen Knoten DUT wurde genutzt. Das Fahrzeug wurde auch als DUT genannt. DUT in Car2x Szenario brauchte auch ein Zertifikat, aber dieses Zertifikat schön für die Programmierungen des DUTs, die die Planung der Fahrzeugbewegungsroute enthält. (Siehe Abb. 82)

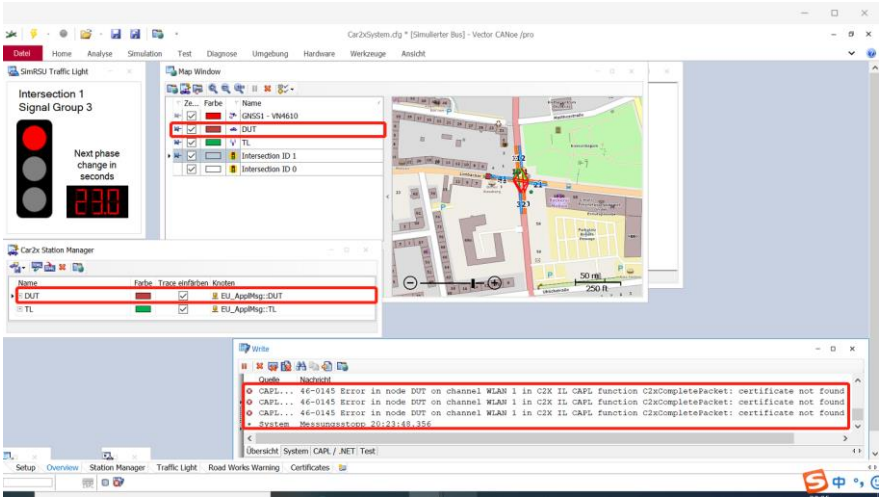


Abbildung 82: Cerzificate not found in Versuch3

Außerdem sollte es Vorsichtig sein, dass Security Layer nicht falsch gewählt wird. (Siehe Abb. 83)

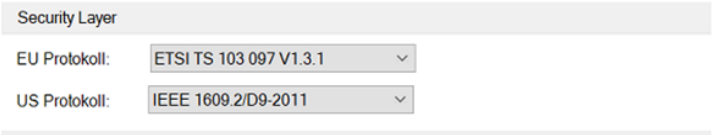


Abbildung 83: Security Layer

Falls falsche Protokoll gewählt wurde, gab es Fehler auf Write-Fenster. Security Header version mismatch.

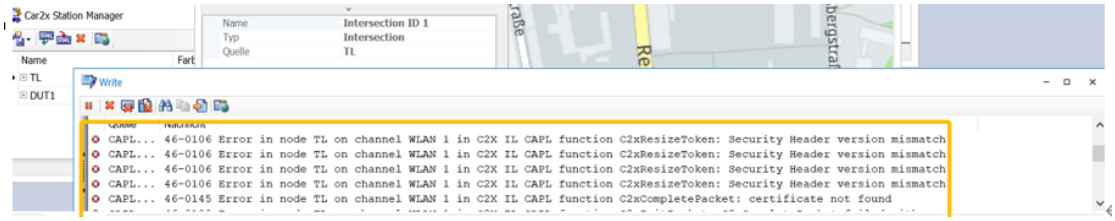


Abbildung 84: Fehler in node TL

5 Zusammenfassung und Ausblick

In diesem Kapitel wird die Arbeit über die Zusammenfassung über Praktikumsversuchen dargestellt. Im Abschnitt Ausblick wird dargestellt, welche weiteren zukünftige Forschungsmöglichkeiten durch diese Ausarbeitung gegeben sind.

5.1 Zusammenfassung

Auf der ersten Praktikumsversuchen geht es darum zwei Fahrzeuge (StudentA und StudentB), miteinander zu kommunizieren. Die beide Fahrzeuge senden und empfangen CAM-Botschaft, die Latitude, Longtitude, Geschwindigkeit und den Abstand zwischen StudentA und StudentB enthält.

Die Route der Fahrzeuge und zwei Fahrzeuge werden durch Car2x Szenario Editor einfach entworfen. In welcher Sekunde werden die Fahrzeuge abfahren oder anhalten, ist die Fahrzeuge direkt in den Griff. Das ist gut geeignet für Anfänger. Die Geschwindigkeit, Beschleunigung, Positionen und Fahrstrecke sind anschaulich für Entwerfer, aber für Benutzer schwer zu sehen. Es muss durch Automatisierung Botschaft zu erstellen, die genaue Wert weitergeben, damit es auf Panel scheint. Manchmal passiert es, dass StudentA noch nicht abfährt und StudentB bereits die Reisedaten von StudentA bekommen. d.h. Die Fahrzeuge und Botschaft über genau Wert nicht gleichzeitig erscheinen. Es gibt Verschiebung.

Andererseits gibt es auch eine andere Möglichkeit, ohne Szenario im CAPL-Programm zu programmieren, um Route und Fahrzeuge festzulegen. Für Anfänger ist die Programmierung nicht einfach zu verfügen, aber die Signalübertragung ist sehr genau und pünktlich.

Die zweiten Praktikumsversuchen bezieht sich darauf, dass das Fahrzeug auf der Grundlage von Praktikumsversuchen 1 in einer dringenden Situation befindet. Das ist sehr wichtig in unserem Laben, autonomes Fahren zu realisieren.

Auf der dritten Praktikumsversuchen handelt es sich darum, dass die Fahrzeuge unter Ampelrichtung fahren. Auf der Grundlage von dem Beispiel „Car2x System“ ändert sich die Programmierungen „RSU_TrafficLight.can“. Nach der Korrektur werden große Hilfe bei Erlernen der CAPL-Programmierung bekommt und tiefes Verständnis für Zertifikat gelernt.

5.2 Ausblick

Mit CANoe werden die Fahrzeuge, Route, die Fahrumgebung simuliert. Durch die Programmierung und Car2x Szenario werden die Situationen, z.B. mit anderem Fahrzeuge zu kommunizieren, in Gefahr, der Ampel zu folgen, realisiert. Die Straßensituationen im Real sind viel mehr als diese Situation. In der Zukunft wird die Programmierung lückenlos sein. In den Versuch 1,2,3 gibt es nur maximal 2 Fahrzeuge. Es werden später mehr Fahrzeuge, z.B. drei oder vier Fahrzeuge, in den Versuch sein. Mehr Fahrzeuge kommunizieren miteinander, Position, Geschwindigkeit zu senden. Falls ein Fahrzeug in Notfall ist, werden andere Fahrzeuge Gefahrensignal empfangen und das gefährliche Fahrzeug vermeiden oder anhalten. Und Die Anwendung von Ampeln hält die Fahrzeuge in Ordnung. Mehrere Straßenzustände können kombinieren.

Im Autopilot-Modus ist es möglich, durch die Analyse von Echtzeit-Verkehrsinformationen automatisch die beste Fahrroute auszuwählen, wodurch Staus erheblich gelockert werden. Darüber hinaus kann durch den Einsatz von On-Board-Sensoren und Kamerasystemen die Umgebung erfasst und schnell angepasst werden, um "null Verkehrsunfälle" zu erreichen. Wenn z. B. ein Fußgänger plötzlich erscheint, kann er automatisch auf eine sichere Geschwindigkeit verlangsamen oder anhalten. Die Stufen des autonomen Fahrens würde zu Stufe 5 (fahrerloses Fahren) mit Technikentwicklung bald sein.

Literaturverzeichnis

[1] Prof. Dr.-Ing. Lutz Eckstein: Automatisiertes-Fahren-VDI-Statusreport-Juli-2018.pdf, 07, 2018.

[2] Prof. Dr.-Ing. J. Thomanek: Modul Ausgewählte Kapitel der Elektro- und Informationstechnik, Sensordatenfusion im Fahrzeug, SS 2019 Version 1.0.

[3] Dipl. inf. Robert K Schmidt, Dipl. ing. Tim Leinmüller, Dr. rer. nat. Bert Böddeke: V2XKommunikation, 2008.

[4] H. Winner, S. Hakuli, F. Lotz, C. Singer (Hrsg.): Car2X, Handbuch Fahrerassistenzsysteme, ATZ/MTZ-Fachbuch, DOI 10.1007/978-3-658-05734-3_28, Frühling, 2015.

[5] Claudia Campolo • Antonella Molinaro, Riccardo Scopigno: Vehicular ad hoc Network, Standards, Solutions, and Research, 2015.

[6] CEN ISO 17419: Intelligent transport systems – cooperative systems – classification and management of ITS applications in a global context (Zugriff am 02, 12, 2020).

[7] OSI-Schichtenmodell, Online Quelle: http://www.elektronikkompendium.de/sites/kom/0301201.htm (Abgerufen am 12, 2020).

[8] Syed Faraz Hasan, Nazmul Siddique Shyam Chakraborty: Intelligent Transportation systems, 802.11 based Vehicular Communications, 2018.

[9] CAM-Nachricht, Online Quelle: https://www.itwissen.info/CAM-cooperative-awareness-message-CAM-Nachricht.html (Zugriff am 27, 12, 2020).

[10] ETSI,ITS,DENM, Online Quelle: http://www.etsi.org/deliver/etsi_ts/102600_102699/10263703/01.01.01_60/ ts_10263703v010101p.pdf. (Zugriff am 05, 02, 2021).

[11] Markus Ullmann, Thomas Strobe, Christian Wastebin: Technical Limitations, and Privacy Shortcomings of the Vehicle-to-Vehicle Communication (Zugriff am 02, 02, 2021).

[12] Vector Informatik: CANoe_Car2x_ProductInformation_DE.pdf, Version 11.0, Online Quelle: vector.com (Zugriff am 02, 02, 2021).

[13] Vector Informatik: CANoe_QuickStartExport_DE.pdf, Version 12.0 SP3, Online Quelle: vector.com (Abgerufen am 02, 02, 2021)

[14] Vector Informatik: www.vector.com (Abgerufen am 02, 02, 2021)

[15] Vector Informatik: hb_capl_programming.pdf, Online Quelle: vector.com (Abgerufen am 02, 02, 2021)

[16] Vector Informatik: CANoe_Car2x_ProductInformation_DE.pdf, Version 13.0, Online Quelle: vector.com (Zugriff am 02, 02, 2020)

[17] Gade Kenneth: A Non-singular Horizontal Position Representation. Journal of Navigation. 63 (3): 395–417. doi:10.1017/S0373463309990415. ISSN 0373-4633. (Zugriff am 01, 01, 2021)

[18] Betty XU: China LTE-V2X Communication Standard Progress (Zugriff am 19, 01, 2021)

[19] Amelia C. Regan, Rex Chen: Vehicular Communications and Networks (Zugriff am 02, 02, 2021)

[20] Prof. Dr.-Ing. J. Thomanek: Vorlesungskript Car2Car, Sommersemester, 2021

[21] Shahana Ibrahim, Dileep Kalathil, Rene O. Sanchez and Pravin Varaiya: Estimating Phase Duration for SPaT Messages 01, 2018

[22] Dipl. Ing. Jürgen Weingart: Standardisation of Spat and Map (Zugriff am 02, 02, 2021)

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe. Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Chemnitz,08,03,2021	Xiaoming Zhou
Ort, Datum	Vorname Nachname