technische universität
dortmund

Diplomarbeit

Infrastruktur für die kooperative Situationsanalyse in dezentralen Verkehrsleitsystemen (SOTIS)

Moritz Hofmann

Fakultät für Informatik
Technische Universität Dortmund

21. September 2009

Gutachter:
Prof. Dr.-Ing. Olaf Spinczyk
Prof. Dr. Gerrit Kalkbrenner

Kurzbeschreibung

Verkehrsmeldungen werden aktuell zentral erhoben und beschränken sich hauptsächlich auf Stausituationen auf Autobahnen. Durch die manuelle Auswertung der eingehenden Daten und die Verbreitung über Rundfunk oder Mobilfunk entstehen Verzögerungen und Ungenauigkeiten. Eine mögliche Lösung für die genannten Probleme stellen dezentrale Verkehrsleitsysteme (*SOTIS*) dar. Diese basieren auf einer fahrzeugübergreifende Analyse von Verkehrssituationen (*kooperative Situationsanalyse*). Hierbei werden die Bewegungsdaten der einzelnen Fahrzeuge über Ad-Hoc-Netzwerke kommuniziert und dezentral zu Verkehrsmeldungen abstrahiert. Durch die Nutzung moderner Fahrzeugsensoren lassen sich eine Vielzahl an Verkehrssituationen erkennen, vor denen nachfolgende Fahrzeuge zeitnah gewarnt werden können.

In der vorliegenden Diplomarbeit wird im Rahmen eines umfassenden Open Source SOTIS-Projekts eine Infrastruktur für die kooperative Situationsanalyse entwickelt. Diese ermöglicht die einfache Integration von Verfahren zur kooperativen Erkennung verschiedener Verkehrssituationen in ein SOTIS und die Evaluierung von Verfahren mit Hilfe einer Simulationsumgebung.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

Die über Rundfunk und Mobilfunk verbreiteten Verkehrsmeldungen und Stauinformationen werden aktuell von zentralen Verkehrsleitstellen bereitgestellt. Deren Mitarbeiter werten dazu Daten von an der Fahrbahn installierten Sensoren wie Videokameras und Induktionsschleifen aus, nehmen relevante Meldungen der Polizei und von Verkehrsclubs entgegen und schlussfolgern daraus die aktuelle Verkehrslage. Auf Grund der zentralen Organisation des Systems und einer geringen Anzahl an Sensoren kann nur ein begrenzter Umfang an Informationen verarbeitet werden. Aktuelle Staumeldungen beschränken sich deshalb auf Autobahnen und sind zudem ungenau. Durch die zentrale Verarbeitung und Übertragung im Verkehrsprogramm des Rundfunks besteht zwischen Auftreten von Ereignissen und der Ankunft der Meldung beim Verkehrsteilnehmer eine Verzögerung von 20 bis 50 Minuten. Auch neuere, digitale Übertragungsverfahren mittels RDS-TMC ([1]) oder Mobilfunk verkürzen diese Zeitspanne nur unwesentlich.

Als mögliche Lösung dieses Problems werden aktuell dezentral organisierte Verkehrsleitsysteme erforscht, die fahrzeuglokale Sensoren zur Erfassung der Bewegungsdaten nutzen. Als Kommunikationsplattform dienen Ad-Hoc-Netzwerke, die unter Nutzung weiterer vekehrsteilnehmender Fahrzeuge diese Daten und daraus generierte Verkehrsinformationen austauschen. Verkehrsleitsysteme dieser Art werden als *SOTIS* (Self Organizing Traffic Information System)[1] bezeichnet. Grundlegende Konzepte dazu wurden zum Beispiel in den Forschungsprojekten NOW – Network-On-Wheels ([3]) und FleetNet ([4]) erarbeitet. Daneben existieren in Teilbereichen bereits erste kommerzielle Lösungen. Um die Probleme proprietärer Lösungen zu vermeiden und eine einheitliche Standardisie-

1 Der Begriff wurde von Lars Wischof et. al. im FleetNet Projekt eingeführt, siehe [2].

rung in diesem Bereich voran zu bringen, soll ein Open Source SOTIS (*OSSOTIS*) entwickelt werden, das eine Architektur und offene Schnittstellen für ein selbstorganisierendes Verkehrsleitsystem zur Verfügung stellt.

1.2 Zielsetzung

In einem dezentralen Verkehrsleitsystem stehen durch die Nutzung der fahrzeuglokalen Sensoren (z.B. GPS, Bremssysteme oder Regensensoren) jedes teilnehmenden Fahrzeugs sehr viele Bewegungsdaten zur Verfügung. Diese müssen kommuniziert und ausgewertet werden. Mittels kooperativer Situationsanalyse lassen sich diese Daten lokal in den einzelnen Fahrzeugen individuell interpretieren, akkumulieren und zu Verkehrsinformationen abstrahieren. Neben Staus lassen sich so auch sicherheitsrelevante Situationen erkennen. Vor dem Hintergrund eines umfassenden OSSOTIS-Projektes zielt diese Arbeit darauf ab, eine Infrastruktur zur Nutzung verschiedener Verfahren zur kooperativen Situationsanalyse zu entwickeln.

Dazu wird festgelegt, welche Daten erhoben und übertragen werden können. Eine lokale Wissensdatenbank ist zu verwalten. Offene Schnittstellen sind zu spezifizieren, um die entwickelten Methoden in ein OSSOTIS zu integrieren und damit die Anbindung an benötigte Komponenten sowie eine darüber liegende Präsentationsschicht (Benutzeroberfläche) zu ermöglichen. Diese Schnittstellen sind so zu gestalten, dass zukünftig verschiedene Verfahren zur kooperativen Situationsanalyse einsetzbar sind. Als konkretes Beispielverfahren wird die Erkennung von Stausituationen gewählt und exemplarisch umgesetzt.

Das beschriebene System wird prototypisch in einer Simulationsumgebung implementiert. Als Programmierumgebung wird aller Voraussicht nach Java verwendet. Die Simulationsumgebung soll es ermöglichen, einzelne Verfahren außerhalb des OSSOTIS-Gesamtsystems vorab zu evaluieren, da die Installation und Aktualisierung der Verfahren in den beteiligten Fahrzeugen eines dezentralen Verkehrsleitsystems sehr aufwändig ist. Dazu werden Testreihen in Form von Verkehrsszenarien verwendet, mit deren Hilfe die Zuverlässigkeit von Verfahren zur kooperativen Situationsanalyse getestet und optimiert werden können.

1.3 Aufbau der Arbeit

Kapitel 2 stellt den aktuellen Stand der Forschung im Bereich dezentrale Verkehrsleitsysteme und den dafür notwendigen technischen Hintergrund im Hinblick auf die kooperative Situationsanalyse vor. Außerdem wird Grundlagenwissen zur Verkehrssimulation vermittelt und vorhandene Simulationsumgebungen werden vorgestellt. Kapitel 3 erläutert das OSSOTIS-Projekt mit den zur Verfügung stehenden Teilkomponenten und die Nutzung der kooperativen Situationsanalyse darin.

Auf Basis dieses Wissens erfolgt in Kapitel 4 die Spezifikation und der Entwurf des Systems. Zuerst werden die Anforderungen und die Realisierung der kooperativen Situationsanalyse als Teil der OSSOTIS-Architektur beschrieben, anschließend erfolgt die Spezifikation der Simulationsumgebung und der Entwurf des Beispielverfahrens zur Erkennung von Stausituationen.

Kapitel 6 beschreibt die prototypische Implementierung und erläutert die Besonderheiten und dabei aufgetretene Probleme. In Kapitel 7 ist ein abschließendes Fazit und ein Ausblick auf die weitergehende Entwicklung zu finden.

2 Grundlagen

2.1 Überblick

In diesem Kapitel werden die Grundlagen erläutert, die für das Verständnis des entwickelten Systems erforderlich sind. In den Abschnitten 2.2 und 2.3 werden zuerst die für dezentrale Verkehrsleitsysteme notwendigen Techniken vorgestellt: fahrzeuglokale Sensoren und die fahrzeugübergreifende Kommunikation mittels Ad-Hoc-Netzwerken. Anschließend wird ein Überblick über bereits existierende Ansätze für dezentrale Verkehrsleitsysteme gegeben, wobei hier der Fokus auf deren Lösungen zur kooperativen Situationsanalyse liegt. Abschnitt 2.5 beschreibt den aktuellen Stand der Technik im Bereich der Verkehrssimulationen. Die einzelnen Themenbereiche werden nur ansatzweise im für die vorliegende Arbeit notwendigen Umfang behandelt, für detaillierte Erläuterungen wird auf die jeweils angegebene Literatur verwiesen.

2.2 Sensoren im Fahrzeug

„Sensoren setzen eine physikalische oder chemische (meist nicht elektrische) Größe Φ in eine elektrische Größe *E* um [...]."[2] Sie bilden also die Schnittstelle zwischen physikalisch bzw. mechanisch erfassten Vorgängen und digitaler Signalverarbeitung. In modernen Kraftfahrzeugen werden eine Vielzahl an verschiedenen Sensoren verwendet, deren Signale über Anpassschaltungen in eine standardisierte, für die weitere Verarbeitung erforderliche Form gebracht werden. Die in Fahrzeugen eingesetzten Sensoren lassen sich nach [6] in drei Anwendungsbereiche gruppieren:

2 [5], S. 110

- Fahrzeugantrieb
- Fahrsicherheit
- Komfort der Insassen

Sensoren im Antriebsbereich werden hauptsächlich für die Motorsteuerung benutzt. Sensoren für die Fahrsicherheit erkennen gefährliche Situationen und ermöglichen die Beeinflussung des Fahrverhaltens durch Bremsen oder Ausweichen, den Schutz der Insassen (v.a. bekannt durch Airbags) oder die Anzeige sicherheitsrelevanter Hinweise. Tabelle 1 zeigt einige Beispiele und ordnet sie dem jeweiligen Anwendungsbereich zu. Diese Zuordnung ist nicht immer eindeutig, insbesondere neuere Sensoren verknüpfen die Bereiche Fahrsicherheit und Komfort.

Fahrzeugsensor	Kontext der gemessenen Information	Anwendungsbereich
Lambdasonde	Abgasregelung	Antrieb
Klopfsensor	Motorsteuerung	Antrieb
Bremssensoren	Bremskraft	Antrieb
ABS-Sensor	Antiblockiersystem	Sicherheit
Airbag-Sensor	Auslösen des Airbags	Sicherheit
Frontradar	Abstand zu voraus fahrenden Fahrzeugen	Sicherheit
Regensensor	Regenintensität	Sicherheit, Komfort
Lichtsensor	Helligkeit der Umgebung	Komfort
Temperatursensor	Umgebungstemperatur	Komfort
Geschwindigkeitssensor	Aktuelle Geschwindigkeit	Komfort
GPS-Sensor	GPS-Position des Fahrzeugs	Komfort

Tabelle 1: Beispiele für Fahrzeug-Sensoren[3]

Für Verkehrsleitsysteme sind besonders die Sensoren der Bereiche Komfort und Fahrsicherheit von Bedeutung. Neben den existierenden, in vielen Fahrzeugen vorhandenen Sensoren werden zunehmend auch Sensoren entwickelt, die sich auf das Umfeld des Fahrzeugs beziehen, beispielsweise mittels Radar oder optischen Sensoren. Die Robert Bosch GmbH prognostiziert dazu: „Dies schließt auch Sensorik ein, die die Bewegung (Kinematik) des Fahrzeugs als Ganzes und als bewegliches Teil im Verkehrsstrom auf der

3 Siehe [7], S. 11 und [5], S. 110 ff.

Straße und den direkten Kontakt der Räder zur Fahrbahn erfassen.“[4] Neue Sensoren werden vor allem in Abhängigkeit von der Preisklasse und der Ausstattung des Fahrzeugs verbaut, so dass die vorhandene und für eine kooperative Situationsanalyse verfügbare Sensorik von Fahrzeug zu Fahrzeug stark variieren kann. [5]

2.3 Fahrzeug-Ad-Hoc-Netzwerke

> „Ad-Hoc-Netzwerke kommen ohne jegliche Infrastruktur aus, insbesondere ohne eine ausgezeichnete Basisstation, welche den Medienzugriff zentral steuert. Diese Netzvariante erlaubt die spontane, nicht vorab geplante Kommunikation zwischen mobilen Endgeräten, wobei einige oder alle Endgeräte auch Daten von anderen Endgeräten weiterleiten können.“[5]

Ad-Hoc-Netzwerke eigenen sich demnach für den dynamischen Datenaustausch zwischen gleichberechtigten Kommunikationspartnern an beliebigen Orten. Gegenüber den Knoten in Infrastrukturnetzen übernehmen die Knoten in Ad-Hoc-Netzwerken mehr Aufgaben und müssen alle Mechanismen zum Medienzugriff enthalten. Dafür sind Ad-Hoc-Netzwerke leichter zu skalieren und deutlich robuster. Eine spezielle Form von Ad-Hoc-Netzwerken stellen mobile Ad-Hoc-Netzwerke, sogenannte *MANETs*, dar. Hierbei bewegen sich die Netzknoten und ändern somit die Struktur des Ad-Hoc-Netzwerkes fortlaufend. Fahrzeug-Ad-Hoc-Netzwerke sind eine spezielle Form von MANETs, bei denen die Netzknoten aus Fahrzeugen gebildet werden. Nachfolgend werden diese auch als *VANET* (vehicular ad-hoc networks) abgekürzt.

MANETs stellen besondere Anforderungen an die verwendete Funktechnik, da sich die Struktur des Netzwerks spontan und häufig ändern kann, wenn einzelne Knoten nicht mehr teilnehmen oder den Empfangsbereich verlassen. Darüber hinaus bewegen sich die Netzknoten eines VANETs typischerweise mit vergleichsweise hohen Geschwindigkeiten, außerdem kann die Anzahl der Knoten, beispielsweise im städtischen Verkehr, sehr hoch sein. Infolgedessen sind effiziente Zugriffsverfahren und ggf. Routingprotokolle für VANETs noch wichtiger als für herkömmliche (mobile) Ad-Hoc-Netzwerke.[6] Stand der Technik bei der Funkübertragung ist der noch in der Entwicklung befindliche Standard *IEEE 802.11p*, eine Weiterentwicklung der bekannten WLAN-Standards der 802.11-Reihe. 802.11p wurde speziell für die *Car-to-Car* Kommunikation entwickelt und geht auf

4 [5], S. 171

5 [8], S. 533

6 Siehe [4], S. 29 f.

die speziellen Bedürfnisse von VANETs ein. Die Vorteile dieser Weiterentwicklung sind die langjährigen Erfahrungen mit 802.11 WLANs und die Kompatibilität mit existierender Hardware und Infrastruktur. [9]

Neben den in dieser Arbeit behandelten dezentralen Verkehrsleitsystemen eignen sich VANETs auch für andere Anwendungen, beispielsweise im Telekommunikations- oder Unterhaltungsbereich. Mögliche Anwendungen hat Wilfried Enkelmann als Teil des *FleetNet*-Projekts in [10] vorgestellt.

2.4 Dezentrale Verkehrsleitsysteme

2.4.1 Überblick

Dezentrale Verkehrsleitsysteme stellen ein wichtiges Anwendungsgebiet von VANETs dar. Sie nutzen das zu Grunde liegende Ad-Hoc-Netzwerk, um verkehrsbezogene Informationen zwischen den Fahrzeugen auszutauschen und diese zu möglichst genauen Informationen über die gegenwärtige Verkehrssituation zu veredeln. Dieser Vorgang der Auswertung der Fahrzeugdaten wird auch als *kooperativer Situationsanalyse*[7] bezeichnet, auf die im nächsten Abschnitt eingegangen wird. [12]

Die danach folgenden Abschnitte stellen abgeschlossene und aktuelle Forschungsarbeiten im Bereich der dezentralen Verkehrsleitsysteme in chronologischer Reihenfolge vor. Dabei werden die Besonderheiten bzw. neuen Erkenntnisse hervorgehoben und wenn möglich auf vorhandene Details zur kooperativen Situationsanalyse eingegangen. Weitere VANET-Projekte, die nicht speziell auf dezentrale Verkehrsleitsysteme ausgerichtet sind sondern z.B. auf einer Infrastruktur basieren oder andere Anwendungen in den Vordergrund stellen, finden sich in [13] und [14]. Erste VANET-Grundlagen wurden bereits Ende der 1980er Jahre im europäischen Forschungsprojekt *Prometheus* ([15]) bearbeitet.

2.4.2 Kooperative Situationsanalyse

Die kooperative Situationsanalyse nutzt, im Gegensatz zur *individuellen Situationsanalyse*, sowohl selbst gesammelte als auch von anderen Fahrzeugen empfangene Fahrzeugdaten zur Erkennung von Verkehrssituationen. Florian Dötzer et. al. klassifizieren diese beiden Arten der Datenaggregation in [16] als „*Single Node Detection*“ bzw. „*Multi Node Detecti-*

7 Im Englischen „*cooperative situation analysis*“, im Gegensatz zur „*individual situation analysis*“. [11], S. 15 ff.

on“. Die Einbeziehung fahrzeugübergreifender Daten erlaubt dabei die zusätzliche Erkennung weiterer Verkehrssituationen, die fahrzeuglokal nicht erkannt werden können (z.B. Stausituationen). Das Beispiel in Abbildung 1 stellt den Vorteil der kooperativen Situationsanalyse heraus. Durch die Kombination der von anderen Fahrzeugen erkannten Informationen „Regen“ und „Schleudergefahr“ kann der Fahrer des nachfolgenden Fahrzeugs im Vorfeld vor der Gefahrensituation gewarnt werden.

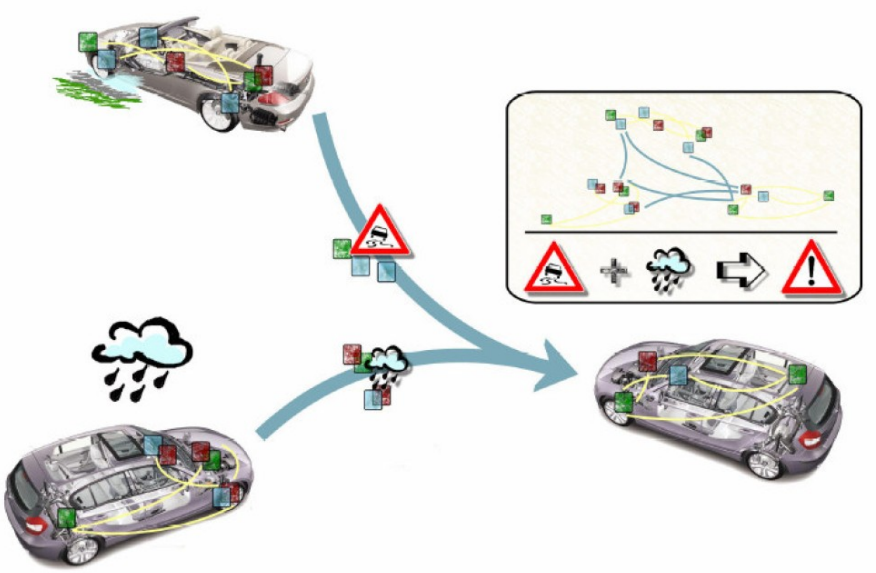

Abbildung 1: Beispiel für die kooperative Situationsanalyse [12]

Nach [11] erschweren zwei Bedingungen die Situationsanalyse: *indirekte Messungen* und *ungenaue Daten.*

- **Indirekte Messungen:** Viele Informationen können nicht direkt von Sensoren gemessen werden, wie es bei einem Regensensor der Fall ist, sondern müssen aus meist mehreren gemessenen Werten interpretiert werden. Beispielsweise kann auf Glätte geschlossen werden, wenn sich die Geschwindigkeit trotz Bremsen nicht verringert. Dasselbe gilt verstärkt für die kooperative Situationsanalyse, da hier deutlich mehr Sensorwerte aggregiert werden.
- **Ungenaue Daten:** Die für dezentrale Verkehrsleitsysteme benutzten Sensoren wurden in der Regel für andere Zwecke verbaut und nach deren Anforderungen konzipiert. Infolgedessen ist die Messgenauigkeit und Verlässlichkeit der Daten für eine präzise Bestimmung der vorliegenden Verkehrssituation möglicherweise nicht ausreichend.

Beide Probleme können durch eine große Anzahl an verfügbaren Sensordaten und möglichst intelligente Algorithmen zur Mustererkennung vermindert werden. Der Entwurf effizienter Verfahren zur kooperativen Situationsanalyse für bestimmte Verkehrssituationen ist nicht Teil dieser Arbeit. Erste Ansätze dazu unter Verwendung von Bayes'schen Netzen wurden in [17] vorgestellt, weitere Herangehensweisen nutzen Entscheidungsbäume oder neuronale Netze. [11]

2.4.3 Inter-Vehicle Hazard Warning

Das deutsch-französische Forschungsprojekt *Inter-Vehicle Hazard Warning* (*IVHW*) wurde 2001 unter Beteiligung der Automobilbranche gegründet. In diesem Projekt wurde ein System zur Übertragung von Warnnachrichten in VANETs spezifiziert. Die Meldungen werden von den beteiligten Fahrzeugen per Broadcast verteilt. Anfangs waren nur vier Typen von Meldungen für Gefahrenstellen vorgesehen, eine mögliche Erweiterung bezieht auch andere Verkehrssituationen mit ein. Für diese Warnmeldungen wurde ein passendes Datenformat entwickelt. Fahrzeuge, die die Meldungen empfangen, überprüfen diese an Hand der eigenen Informationen und bewerten die Relevanz für das eigene Fahrzeug. Hierfür wurden Algorithmen entwickelt, die entscheiden, ob eine Warnung dem Fahrer übermittelt werden soll. [18]

2.4.4 Invent – VLA

Die *Verkehrsleistungsassistenz* (*VLA*) ist ein Teilprojekt der Forschungsinitiative *INVENT* (*Intelligenter Verkehr und nutzergerechte Technik*), die zwischen 2001 und 2005 mit verschiedenen Ansätzen den Problemen des steigenden Verkehrsaufkommens begegnen wollte. Das Teilprojekt VLA hat sich als Ziel die Optimierung des Verkehrsflusses gesetzt, wobei der Schwerpunkt auf dem außerstädtischen Verkehr liegt. Die beschriebenen Lösungsansätze zur schnelleren Auflösung von Staus benötigen genaue Daten über das Stauende, um die Reaktionszeit des Fahrers durch geeignete Informationen zu verkürzen. Hierbei wird auf dezentrale Verkehrsleitsysteme zurückgegriffen:

> „Zum anderen können die Daten auch durch entsprechend ausgerüstete Fahrzeuge selbst erhoben werden, etwa mit „Floating Car Data“ (FCD). Durch den Aufbau und Einsatz geeigneter Kommunikationswege und -strategien werden daraus die erforderlichen Informationen zentral oder dezentral gesammelt und an die Fahrzeuge weitergeleitet.“ [19]

2.4.5 Vehicle Safety Communications

Das *Vehicle Safety Communications* (*VSC*) Projekt startete 2002 mit dem Ziel, die Sicherheit in Fahrzeugen mittels Fahrzeugkommunikation zu erhöhen. Dazu wurden mögliche Vorteile von Ad-Hoc-Netzwerken und Infrastruktur-Netzwerken für Sicherheitsanwendungen identifiziert und die Anforderungen an die Kommunikation herausgearbeitet. Im Bereich der Ad-Hoc-Netzwerke wurden kooperative Systeme als mögliche Anwendung des VANETs vorgestellt, aber technisch noch nicht behandelt. [20]

2.4.6 FleetNet

Im *FleetNet*-Forschungsprojekt wurde von 2000 bis 2003 von mehreren Firmen der Automobilbranche und dem Fraunhofer-Institut für Offene Kommunikationssysteme (FOKUS) unter Leitung der Daimler AG Grundlagenarbeit im Bereich der Fahrzeugkommunikation geleistet. Die Nutzbarkeit von Ad-Hoc-Netzwerken auch bei geringer Fahrzeugabdeckung wurde nachgewiesen und ein positionsbasiertes Routingverfahren innerhalb des VANETs wurde beschrieben. Ein weiterer Schwerpunkt lag auf der Entwicklung eines dezentralen Verkehrsleitsystems als Anwendungsbeispiel für VANETs, hierfür wurde der Begriff *SOTIS* eingeführt. Das SOTIS-Konzept war der erste Prototyp dieser Art und bildet die Basis einiger nachfolgender Forschungsarbeiten. Dabei wurden neben der Motivation für dezentrale Verkehrsleitsysteme deren Anforderungen herausgearbeitet. Der Fokus wurde auf die Lösung technischer Probleme bei der Funkübertragung und die Systemarchitektur gelegt. Die SOTIS-Architektur enthält einen *SOTIS-Core*, der für die kooperative Situationsanalyse zuständig ist. Außerdem werden folgende weitere Komponenten benötigt: Eine Wissensdatenbank zur Verwaltung der Fahrzeugdaten, Kartenmaterial, Adapter für Sensoren und die Positionsbestimmung sowie eine Komponente für die Funkübertragung. Abbildung 2 stellt die Architektur als Blockdiagramm dar.

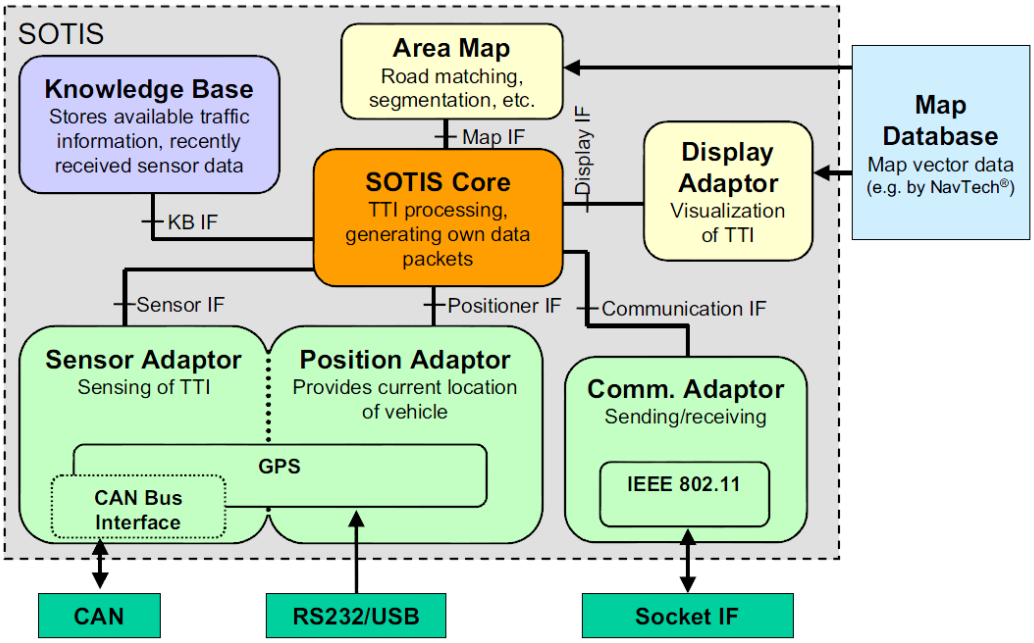

Abbildung 2: SOTIS-Architektur im FleetNet-Projekt [21]

Zur Steuerung der Kommunikation nutzt dieses SOTIS eine *Segment-Oriented Data Abstraction und Dissemination* (*SODAD*) genannte Technik, die die Straßen des digitalen Kartenmaterials in Segmente variabler Länge einteilt und die räumliche Verteilung der Datenpakete mittels der zwei Möglichkeiten Funkübertragung oder Transport im Fahrzeug beschreibt. Auf den Ablauf und die Struktur der kooperativen Situationsanalyse wurde nicht eingegangen, sie wurde lediglich als „Black Box“-Komponente in Form des SOTIS-Core in die Architektur integriert. [22], [4]

2.4.7 Network on Wheels

Das 2008 beendete Projekt *Network on Wheels* (*NOW*) wurde 2004 als Nachfolger des Fleetnet-Projekts gestartet. Die Ziele waren die Entwicklung von Kommunikationsprotokollen für Ad-Hoc-Netzwerke zur Gefahrenwarnung, Verkehrsführung und für Infotainment-Angebote und die Erforschung von Sicherheitskonzepten in VANETs. Die Ergebnisse wurden zur Standardisierung dem *CAR 2 CAR Consortium* bereitgestellt. Neben der Behandlung technischer Fragestellungen wurden auch erste Strategien zur Markteinführung der entwickelten Systeme hervorgebracht. Aufbauend auf die grundlegenden Ergebnisse des FleetNet-Projekts wurde eine offene Kommunikationsplattform auf Basis der IEEE 802.11 Funktechnologie entwickelt, die die genannten Anwendungen unterstützt und die Grundlage weiterer Projekte bilden soll. [23]

Die kooperative Situationsanalyse zur Gefahrenerkennung wurde als eine mögliche Anwendung der NOW-Plattform beschrieben und gegenüber den Ergebnissen des FleetNet-Projekts genauer spezifiziert. Abbildung 3 zeigt die Vorgänge als sechsstufigen Kreislauf: Von Sensoren im Fahrzeug gemessene Werte (1) werden ausgewertet (2) und im VANET verbreitet (3). Eingehende Daten werden mittels kooperativer Situationsanalyse erneut ausgewertet (4) und die Ergebnisse an Hand von bisherigen Daten und eigenen Sensorwerten verifiziert (5). Daraus lassen sich fahrzeuglokal relevante Gefahrenstellen vorhersagen. Die fahrzeugübergreifende Kommunikation wird über ein relevanzbasiertes Konzept gesteuert, dass den einzelnen Informationen Werte zuordnet, die unter Einbeziehung des Fahrzeugkontextes, des Nachrichteninhalts und der Auslastung des Netzwerks berechnet werden. Damit werden wichtige Informationen bevorzugt verschickt und unwichtige Informationen (Relevanz unter einem bestimmten Wert) belasten das Netzwerk nicht. [24]

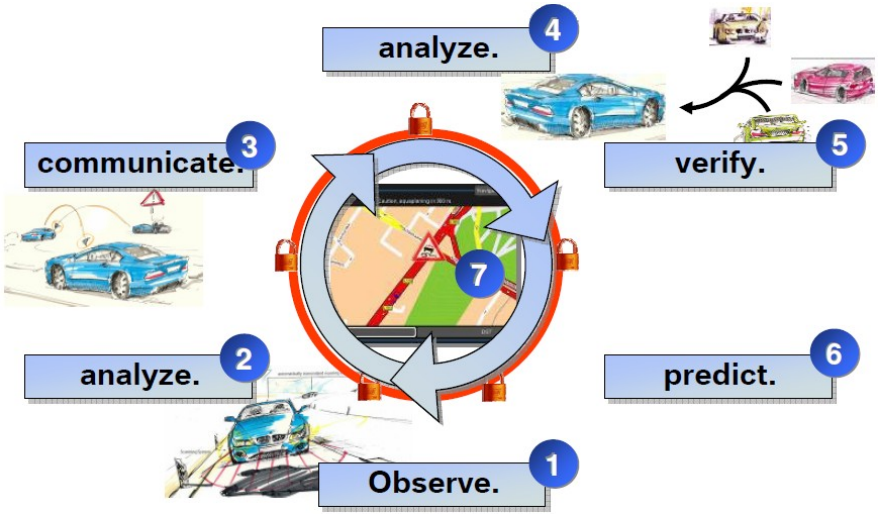

Abbildung 3: Kooperative Situationsanalyse im NOW-Projekt [24]

2.4.8 PreVENT – WILLWARN

PReVENT ist ein Projekt der europäischen Automobilhersteller unter Förderung der EU-Kommission. Es hat sich 2004 zum Ziel gesetzt, Verkehrsunfälle durch präventive Sicherheitsanwendungen zu verhindern. *WILLWARN* (*Wireless Local Danger Warning*) ist ein Teilprojekt zur frühzeitigen Warnung vor Gefahrenstellen. Die VANET-Anwendung gliedert sich in die drei Bereiche Gefahrenerkennung, fahrzeuglokale Verarbeitung der Warnung und dezentrale Verteilung der Warnung. Zur Kommunikation wird der IEEE-Standard 802.11p benutzt. WILLWARN nutzt bestehende Fahrzeugsensoren, um die benötigte Ausstattung gering zu halten. [25]

Das WILLWARN-System besteht aus vier Modulen. Das *Hazard Detection Module* verarbeitet die Sensordaten und erkennt darin Gefahrenstellen. Im *Warning Management Module* werden die vom Hazard Detection Module und über das VANET eingegangen Warnmeldungen ausgewertet und die Relevanz für das eigene Fahrzeug bewertet. Außerdem werden in diesem Modul neue Warnmeldungen generiert und mit der GPS-Position verknüpft. Die Fahrzeug-zu-Fahrzeug Kommunikation übernimmt das *Communication Module*, das *Hazard Warning Module* ist für die Anzeige der Warnmeldungen im Fahrzeug-Cockpit verantwortlich. Abbildung 4 zeigt den Aufbau des WILLWARN-Systems und das Zusammenspiel der einzelnen Module. [25]

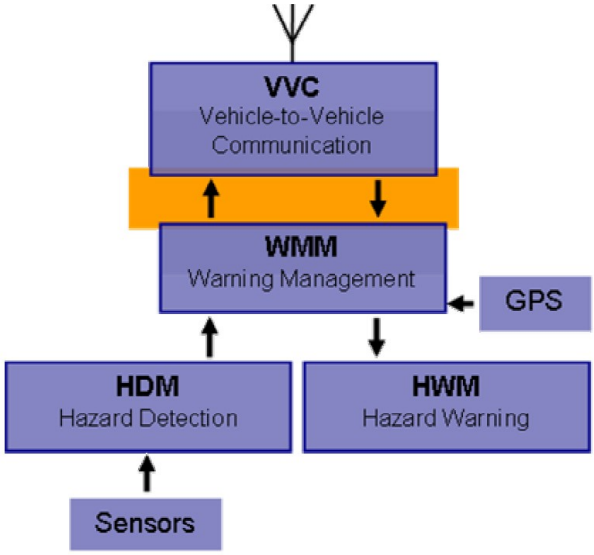

Abbildung 4: Aufbau des WILLWARN-Systems [25]

2.4.9 Safespot

In dem von Fiat geleiteten Projekt *Safespot* mit 51 Partnern aus der Automobilbranche und Forschungseinrichtungen soll von Anfang 2006 bis Ende 2009 ein *Safety Margin Assistant* entwickelt werden, der im Vorfeld Gefahrensituationen erkennt und dem Fahrer sicherheitsrelevante Informationen aus dem Fahrzeugumfeld zur Verfügung stellt. Dieses intelligente System greift neben der Car-to-Car Kommunikation auch auf Elemente der Infrastruktur zurück, um Gefahren kooperativ zu erkennen. Infrastrukturknoten nutzen dabei die selbe Technik wie die Fahrzeuge, so dass sie sich von diesen nur durch den festen Standort unterscheiden. Dies reduziert die Komplexität für die Einbeziehung in das VANET. Auch in diesem Projekt wird der IEEE Standard 802.11 als Funktechnologie verwendet. [26]

Gefahrenquellen werden bei Safespot in zwei Klassen eingeteilt. *Static Black Spots* sind dauerhafte, zur Verkehrsinfrastruktur gehörende Gefahren wie Tunnel oder enge Kurven. Vor diesen sollen hauptsächlich Infrastrukturknoten warnen. Als *Dynamic Black Spots* werden die auch in anderen Forschungsprojekten thematisierten Gefahrensituationen bezeichnet, zum Beispiel Stausituationen, Unfälle oder ähnliches. Die Informationen werden in beiden Fällen unter den teilnehmenden Fahrzeugen im VANET verbreitet und sollen fahrzeuglokal validiert werden. [26]

2.4.10 CAR 2 CAR Communication Consortium

Das 2007 gegründete *CAR 2 CAR Communication Consortium* (*C2CC*) strebt einen europaweiten, einheitlichen Standard für VANETs und darauf aufbauender dezentraler Verkehrsleitsysteme an. Das Ziel ist die herstellerübergreifende und fahrzeugtypunabhängige Einsetzbarkeit. Die technische Grundlage dafür bildet die WLAN-Funktechnologie. Das C2CC unterstützt eine Reihe von Forschungsprojekten und bindet deren Ergebnisse in den Standardisierungsprozess ein. [27]

2.4.11 Sichere Intelligente Mobilität: Testfeld Deutschland

Das Ende 2008 mit einer geplanten Laufzeit von vier Jahren angelaufene Projekt *Sichere Intelligente Mobilität: Testfeld Deutschland* (*simTD)* hat sich zum Ziel gesetzt, Fahrzeugkommunikation in einem Feldversuch auf die Nutzbarkeit in der Praxis zu untersuchen. „simTD ist ein Gemeinschaftsprojekt führender deutscher Automobilhersteller, Automobilzulieferer, Kommunikationsunternehmen und Forschungsinstitute“ ([28]) und wird von mehreren Bundesministerien gefördert. Neben der Kommunikation zwischen Fahrzeugen (*Car-to-Car*) wird auch die Kommunikation zwischen Fahrzeugen und stationären Verkehrszentralen (*Car-to-X*) einbezogen. Für den Feldversuch ausgewählt wurden folgende Funktionen: Erfassung der Verkehrslage, Verkehrsfluss-Information, Verkehrsfluss-Steuerung, lokale Gefahrenwarnung (für Hindernisse, Stausituationen, Wetter und Einsatzfahrzeuge), Fahrerassistenz, Internetzugang und lokale Informationsdienste. Die erarbeiteten Ansätze sollen in einem Testfeld in Hessen mit 400 Fahrzeugen und 150 stationären Einheiten unter realen Bedingungen erprobt werden. [28]

Bei diesem Projekt ist demnach unter anderem mit Erfahrungen bei dem Einsatz der kooperativen Situationsanalyse unter realen Bedingungen zu rechnen. Insbesondere das Verhalten bei einer großen Fahrzeugabdeckung wurde in vorangegangenen Forschungsprojekten nur in Simulationsumgebungen evaluiert.

2.5 Verkehrssimulation

2.5.1 Überblick

Die Simulation von Straßenverkehr ist für die Evaluation von Forschungsprojekten und das Verstehen komplexer Verkehrssituationen unabdingbar. Trotz der üblichen Einschränkungen und Ungenauigkeiten durch Abstraktion sind Simulationen eine sinnvolle

Alternative zu Untersuchungen im laufenden Verkehr, die sehr aufwändig und wegen möglichen Verkehrsbehinderungen und Sicherheitsrisiken nur für erprobte Konzepte mit hoher Erfolgswahrscheinlichkeit durchführbar sind. Verkehrssimulationen werden bisher hauptsächlich in der Stauforschung und zur Optimierung von Routing-Algorithmen eingesetzt, neuerdings aber auch zur Erprobung von Fahrerassistenz- und Verkehrsleitsystemen. [29]

Der folgende Abschnitt beschreibt die den Simulationen zu Grunde liegenden Verkehrsmodelle. Anschließend wird eine Auswahl aktueller Simulationsumgebungen vorgestellt. Bei der Auswahl wurde der Fokus neben einer weiten Verbreitung des Produkts auf die Möglichkeiten zur Erweiterung der Simulationsumgebung gesetzt, da dies die den Einsatz im OSSOTIS-Projekt und die Anpassung an dessen Anforderungen erleichtert. Daneben existiert eine Vielzahl proprietärer Lösungen, die hauptsächlich von der Automobilindustrie entwickelt und eingesetzt werden. Einen Überblick über weitere Simulationen gibt Panos Pavlidis in [29].

2.5.2 Verkehrsmodelle

> „Modelle und Simulation dienen als Hilfsmittel zum Umgang mit der Realität. Wenn Szenarien in der Wirklichkeit zu groß oder zu komplex sind, zeitlich lang dauern oder zu kostenintensiv sind, um sie im direkten Versuch zu beobachten, dient ein Modell zur Darstellung und/oder Vereinfachung.“[8]

Verkehrsmodelle abstrahieren demnach den Straßenverkehr und bilden ihn vereinfacht ab, um in Simulationen Experimente am dynamischen Verhalten durchführen zu können. Diese lassen sich je nach Qualität des Verkehrsmodells auf den realen Verkehr übertragen. Bei Verkehrsmodellen lassen sich neben der Nutzung von diskreten oder kontinuierlichen Variablen hauptsächlich zwei Typen unterschieden:

- *Makroskopische Modelle* bilden Verkehrsflüsse in großen Straßennetzen nach, ohne einzelne Fahrzeuge zu modellieren. Hierzu werden aggregierte Daten wie die Verkehrsdichte oder die mittlere Geschwindigkeit modelliert. Makroskopische Modelle werden hauptsächlich zur Prognose der Verkehrsdichte eingesetzt, zum Beispiel im *OLSIM*-Projekt[9]. [32], [33]

8 [30], S. 2

9 OLSIM (Online Simulation) visualisiert das aktuelle und prognostizierte zukünftige Verkehrsaufkommen im Autobahnnetz von Nordrhein-Westfalen. OLSIM ist unter *http://www.autobahn.nrw.de* im Internet erreichbar. [31]

- *Mikroskopische Modelle* bilden jedes einzelne Fahrzeug ab und bewegen diese an Hand der jeweiligen Fahrzeugeigenschaften, der Geschwindigkeit und des Umfeldes, gekennzeichnet insbesondere durch das vorausfahrende Fahrzeug. Daher werden derartige Verkehrsmodelle auch als *Fahrzeugfolge-Modelle* bezeichnet. Der Zustand eines Fahrzeugs ist mindestens durch die Position, die Geschwindigkeit und die Beschleunigung gekennzeichnet. [32], [33]

Daneben existieren *mesoskopische Modelle* als eine Art Zwischenstufe zwischen makro- und mikroskopischen Modellen, bei denen einzelne Fahrzeuge vereinfacht modelliert werden und das Verkehrsnetz. Neuerdings wird darüber hinaus noch ein vierter Detaillierungsgrad abgegrenzt, die *submikroskopischen Modelle*. Kai Erlemann definiert diese in seiner Dissertation so: „Eine weitergehende Verfeinerung der mikroskopischen Simulationsebene ist das submikroskopische Modell, das nicht nur einzelne Fahrzeuge auf einem relativ abstrakten Niveau beschreibt sondern darüber hinaus auch akribisch einzelne Funktionsgruppen der Fahrzeugmechanik und -steuerung nachbildet. Jedes einzelne fahrrelevante Bauteil und sein Zusammenspiel mit den übrigen Komponenten wird dabei so detailliert und physikalisch korrekt wie möglich simuliert.“ [34]

In den nächsten Abschnitten werden die wichtigsten Verkehrsmodelle kurz vorgestellt. Zur Simulation der kooperativen Situationsanalyse eignen sich nur mikroskopische Modelle, da die Eigenschaften der einzelnen Fahrzeuge beobachtet und ausgewertet werden müssen. Submikroskopische Modelle sind sehr rechenintensiv und werden bisher kaum in Simulationsumgebungen eingesetzt. Im Folgenden werden deshalb nur Simulationen mit mikroskopischem Modell betrachtet.

Nagel-Schreckenberg-Modell

Kai Nagel und Michael Schreckenberg bilden den Straßenverkehr in dem nach ihnen benannten *Nagel-Schreckenberg-Modell* als so genannter *Zellularautomat* ab. „Dabei werden die Fahrzeuge als diskrete Objekte betrachtet, die sich nach einfachen stochastischen Regeln auf einem Gitter bewegen.“ ([35]) Das Modell nutzt diskrete Werte für Ort und Zeit (Geschwindigkeit) und ist rundenbasiert. Das Gitter, auf dem sich die Fahrzeuge bewegen, besteht aus einzelnen Zellen mit fester Länge, die jeweils genau ein oder kein Fahrzeug enthalten. Der einfachste Fall eines Einspurverkehrs auf einer kreisförmigen Straße lässt sich damit durch vier Regeln modellieren, die für jeden diskreten Zeitschritt nacheinander, aber für alle Fahrzeuge parallel ausgeführt werden:

- „**Beschleunigung:** Alle Fahrzeuge mit Geschwindigkeit v, die kleiner als die Maximalgeschwindigkeit v_{max} ist, beschleunigen um eine Geschwindigkeitseinheit [$v \rightarrow v + 1$].
- **Abbremsen:** Ist der Abstand d eines Fahrzeuges zum vorausfahrenden (d.h. die Zahl der freien Zellen vor dem Fahrzeug) nicht größer als seine momentane Geschwindigkeit v (d.h. $d \leq v$), so wird es auf die neue Geschwindigkeit $d - 1$ abgebremst [$v \rightarrow d - 1$].
- **Trödeln:** Mit einer vorgegebenen Wahrscheinlichkeit p wird die Geschwindigkeit eines fahrenden Fahrzeugs ($v > 0$) um den Wert 1 verringert [$v \rightarrow v - 1$].
- **Fahren:** Jedes Fahrzeug wird um v Plätze weiterbewegt.“ [35]

Das Verhalten eines Fahrzeuges hängt demzufolge hauptsächlich vom Verhalten des vorausfahrenden Fahrzeuges ab. Mit dem Nagel-Schreckenberg-Modell konnte erstmals das spontane Auftreten von Stausituationen erklärt und der Verkehr größerer Straßennetze in Echtzeit simuliert werden. [35]

Die Grundzüge des Nagel-Schreckenberg-Modells wurden in vielen nachfolgenden Forschungsarbeiten erweitert, um auch komplexere Verkehrssituationen (vor allem Mehrspurverkehr) mit Zellularautomaten modellieren zu können. „[Mit dem beschriebenen Zellularautomaten] lassen sich die grundlegenden Phänomene sowohl qualitativ als auch quantitativ gut reproduzieren. Es ist sogar möglich, Mehrspurverkehr mit landesüblichem Spurwechselverhalten nachzustellen [...]. Auch Schnittstellen zwischen verschiedenen Straßentypen und spezielle Geometrien, z.B. Kreisverkehr, sind ohne Probleme einbeziehbar.“ [35]

Optimal-Velocity-Modell

Das *Optimal-Velocity-Modell* ist ein Abstands-Modell, bei dem das Verhalten der Fahrzeuge durch Differentialgleichungen beschrieben wird. Dieses deterministische Verfahren wurde 1995 von Bando et. al. in [36] vorgestellt. Jedes Fahrzeug versucht hierbei seine Geschwindigkeit an eine optimale Geschwindigkeit anzunähern, die vom Straßentyp abhängig ist. Die Grundzüge des Optimal-Velocity-Modells wurden in den folgenden Jahren mehrfach überarbeitet und durch veränderte Differentialgleichungen angepasst, um die Simulationsergebnisse zu verbessern. Dabei konnte das Modell bei geringem bis mittlerem Verkehrsaufkommen realistische Ergebnisse erreichen:

> „Ein direkter Vergleich der mit dem Bando-Modell und seinen Variationen erzielbaren Simulationsergebnisse mit denen anderer Verhaltensmodelle

zeigt, dass für zweispurige Autobahnen die Geschwindigkeits-Dichte-Verhältnisse bis zu einer Verkehrsstärke von ca. 3000 Fz/h gut mit den real beobachtbaren Werten übereinstimmen.“[10]

Fahrzeugfolge-Modell nach Krauß

Das *Fahrzeugfolge-Modell nach Krauß* (auch *SK-Modell*) von Stefan Krauß ist ein Beispiel für Coupled-Map-Modelle. Bei diesen Verkehrsmodellen wird die Zeit als diskrete Variable und das Fahrzeugverhalten durch diskrete Abbildungen (*maps*) modelliert. Im SK-Modell wird jedes Fahrzeug durch die Eigenschaften Maximalgeschwindigkeit, Fahrzeuglänge, Brems- und Beschleunigungsvermögen beschrieben, zu denen folgende Annahmen getroffen werden:[11]

- Die Fahrzeuge fahren kollisionsfrei.
- Die Fahrzeuge halten eine Maximalgeschwindigkeit v_{max} ein.
- Das Bremsvermögen a und das Beschleunigungsvermögen b der Fahrzeuge ist jeweils begrenzt.

In Abhängigkeit der Variablen a und b lassen sich die in dem Modell simulierten Verhaltensmuster der Fahrzeuge in drei Klassen einteilen:

- **Klasse 1:** Modelle mit niedriger Verzögerung und niedriger Beschleunigung.
- **Klasse 2:** Modelle mit hoher Verzögerung und niedriger Beschleunigung.
- **Klasse 3:** Modelle mit hoher Beschleunigung.

Modelle der Klasse 1 ähneln dem Optimal-Velocity-Modell, Modelle der Klasse 2 ähneln dem Nagel-Schreckenberg-Modell. Die Klasse 3 spiegelt nicht das reale Fahrzeugverhalten wieder und ist demzufolge nicht für Simulationen geeignet. Durch die hohe Beschleunigung und damit fehlendes „Trödeln“ treten keine spontanen Stausituationen auf, die in der Regel durch Verkehrsmodelle erklärt werden sollen. [37], [38]

10 [34], S. 52

11 Die aus den Annahmen resultierenden Gleichungen und diskreten Abbildungen sind in [37] beschrieben.

2.5.3 VISSIM

Die Simulation *VISSIM* (*Verkehr in Städten – Simulationsmodell*) wurde an der Universität Karlsruhe entwickelt und wird von der ptv AG in Karlsruhe vertrieben. Es ist die meistgenutzte Verkehrssimulation in Deutschland[12]. VISSIM ist auf den innerstädtischen Verkehr spezialisiert, kann aber auch Autobahnverkehr simulieren. Neben der Simulation von Fahrzeugen wird außerdem die Einbindung von Fußgängern, Fahrrädern und öffentlichen Verkehrsmitteln unterstützt. Externe Anwendungen können über eine COM-Schnittstelle auf VISSIM zugreifen. [34]

Das VISSIM zu Grunde liegende Verkehrsmodell hat Rainer Wiedemann bereits 1974 in seiner Habilitationsschrift ([40]) entwickelt. Bei diesem Fahrzeugfolge-Modell werden physische und psychologische Aspekte der Fahrer berücksichtigt. Der Grundgedanke ist die Annahme, dass ein Fahrer bzw. das von ihm geführte Fahrzeug in einem der folgenden vier verschiedenen Fahrmodi sein kann: freies Fahren, Annäherung, Folgen und Bremsen. Der Wechsel zwischen den einzelnen Modi erfolgt in Abhängigkeit vom Abstand und der Geschwindigkeitsdifferenz zum vorausfahrenden Fahrzeug. Bei mehrspurigen Straßen wird darüber hinaus ein regelbasiertes Spurwechselverfahren angewandt. [41]

2.5.4 CORSIM

CORSIM (*Corridor Traffic Simulation Model*) ist das in den USA am weitesten verbreitete mikroskopische Verkehrssimulationsprogramm. Es wird seit den 1970er Jahren von der Federal Highway Administration (FHWA) entwickelt und vertrieben. CORSIM eignet sich zur Simulation jeglichen Verkehrsaufkommens und liefert sowohl bei geringem Aufkommen als auch bei Stausituationen realistische Ergebnisse. Außerdem ist es möglich, sämtliche geometrischen Gegebenheiten und Verkehrssteuerungsanlagen abzubilden. Zur Kommunikation mit externen Programmen wurde eine spezielle Schnittstelle (*TSIS*) integriert, die es erlaubt, Fahrzeuge zu bewegen, auf deren Eigenschaften zuzugreifen und die Entscheidungslogik zu beeinflussen. [32]

CORSIM nutzt ein Fahrzeugfolge-Modell, das Ähnlichkeiten mit dem in VISSIM verwendeten Modell aufweist. Das Modell besteht allerdings aus zwei Komponenten: *FRESIM* modelliert Autobahnverkehr, *NETSIM* ist ein Modell für Stadtverkehr. Jedes Fahrzeug versucht den Abstand zum vorausfahrenden Fahrzeug bei gleichzeitiger Berück-

12 Siehe [34], S. 14. Nach [39] ist VISSIM auch die weltweit führende Verkehrssimulation, während Owen et. al. in [32] diesen Platz CORSIM zusprechen.

sichtigung der Geschwindigkeitsbegrenzung zu minimieren. Dabei werden zehn Typen von Fahrern unterschieden, die jeweils unterschiedliche Eigenschaften bezüglich Fahr- und Abbiegeverhalten aufweisen. [42]

Als Nachfolger von CORSIM wird derzeit *NGSIM* entwickelt, welches die in CORSIM vorhanden Schwachstellen beseitigen soll. Diese hat Kai Erlemann wie folgt zusammenfasst:

> „[Die] lange Tradition von CORSIM ist aber andererseits ein Nachteil, da viele der verkehrstechnischen und informatischen Grundlagen mittlerweile überholt sind. So leidet CORSIM unter einer veralteten und schwer wartbaren Code-Basis in Fortran, einer nicht mehr konkurrenzfähigen Bedienoberfläche und teilweise nicht mehr zeitgemäßen Modellierungsansätzen.“[13]

2.5.5 SUMO

Das Programmpaket *SUMO* (*Simulation of Urban Mobility*) ist eine Open Source Verkehrssimulation vom Deutschen Zentrum für Luft- und Raumfahrt (DLR) und wird seit 2000 entwickelt. Das Ziel war die Bereitstellung einer frei verfügbaren und portablen Simulationsplattform für große Straßennetze, mit der in Forschungsprojekten Verkehrsmodelle, Fahrzeugverhalten und Routing-Algorithmen einheitlich getestet und verglichen werden können. SUMO nutzt das mikroskopische Fahrzeugfolge-Modell nach Krauß (siehe Abschnitt 2.5.2). Dieses sehr schnelle Modell erlaubt die Simulation von mehr als 100.000 Fahrzeugen in Echtzeit auf einem Desktop-PC[14]. Zur Einbindung externer Anwendungen bietet SUMO eine Schnittstelle mit Client/Server-Architektur (*TraCI, Traffic Control Interface*). Mit TraCI können Parameter in einer laufende Simulation bearbeitet und Fahrzeuginformationen abgefragt werden. SUMO agiert hierbei als Server, an den verschiedene Kommandos per TCP gesendet werden können. TraCI wurde bereits in mehreren Forschungsprojekten zur Erweiterung von SUMO genutzt, z.B. zur Verbindung mit einem Netzwerksimulator oder der Anreicherung des Straßennetzes um Ereignisse und Notizen. [43], [45]

Das Programmpaket SUMO besteht aus den folgenden Anwendungen[15]:

---

13 [34], S. 16

14 Siehe Zusammenfassung in [43] und Vergleich verschiedener Modelle in [44]. Der Vergleich wurde 2004 durchgeführt, demzufolge ist mit einem aktuellen PC eine deutlich höhere Simulationsleistung zu erwarten.

15 Siehe [45] und [46]

- *SUMO* simuliert ein vorgegebenes Verkehrsszenario auf der Kommandozeile. Übergeben werden müssen ein Straßennetz und eine Menge von Routen für die Fahrzeuge.
- *GUISIM* zeigt die Simulation in einer grafischen Oberfläche an. Die darunter liegende Simulationsanwendung ist mit der Kommandozeilenversion identisch, auch die Ein- und Ausgaben stimmen überein.
- *NETCONVERT* importiert Straßennetze zur Nutzung in SUMO bzw. GUISIM.
- *NETGEN* generiert automatisch abstrakte Straßennetze.
- *OD2TRIPS* importiert O/D-Matrizen und teilt sie in einzelne Routen auf.
- *JTRROUTER* erstellt Routen für Fahrzeuge anhand von Gewichten für die Abbiegemöglichkeiten an Kreuzungen.
- *DUAROUTER* erstellt Routen für Fahrzeuge mittels *Shortest-Path-Algorithmus*.
- *DFROUTER* erstellt Routen für Fahrzeuge an Hand von Daten aus Induktionsschleifen.
- *POLYCONVERT* importiert geometrische Daten zur Visualisierung in GUISIM.

3 OSSOTIS-Projekt

Die vorliegende Arbeit behandelt einen Teilbereich eines umfassenden Open Source SOTIS Projekts (OSSOTIS). Der folgende Abschnitt beschreibt die Ziele und Anforderungen an die OSSOTIS-Plattform und grenzt sie damit von den existierenden SOTIS-Konzepten ab. Anschließend werden die für die OSSOTIS-Architektur benötigten Komponenten vorgestellt und darauf eingegangen, welche Funktionen diese jeweils übernehmen.

3.1 Ziele von OSSOTIS

Die in Abschnitt 2.4 vorgestellten Projekte setzen an vielen Stellen proprietäre Lösungen ein und werden infolgedessen Probleme bei der notwendigen Interkonnektivität aufweisen. Die Vorteile der dezentralen Kommunikation können die Systeme nur nutzen, wenn möglichst viele Teilnehmer untereinander kommunizieren können und nicht mehrere, voneinander abgeschottete Systeme parallel existieren. Zentrales Ziel des OSSOTIS-Projekts ist es deshalb, die einheitliche Standardisierung im Bereich der dezentralen Verkehrsleitsysteme durch die Entwicklung eines Open Source SOTIS voran zu bringen. Dazu sollen eine Architektur und offene Schnittstellen zur Verfügung gestellt werden. Das zu entwickelnde System soll ähnlich wie die bestehenden SOTIS-Lösungen Sensordaten der teilnehmenden Fahrzeuge über ein Ad-Hoc-Netzwerk austauschen und dezentral zu Verkehrsinformationen veredeln. Zur Kommunikation werden bekannte Funktechniken (WIFI) genutzt.

3.2 OSSOTIS-Architektur

Die OSSOTIS-Architektur wird parallel zu dieser Arbeit von Christopher Schaumann entwickelt ([47]), deshalb beziehen sich die folgenden Ausführungen auf eine vorläufige Version der Architektur. Die beschriebenen Funktionen der Komponenten sind für das SOTIS notwendig, allerdings könnte deren Verteilung auf die einzelnen Komponenten noch verändert werden. Diese Arbeit behandelt nur einen Teilbereich des OSSOTIS-Projekts und hat damit keinen Einfluss auf die Gesamtarchitektur.

Abbildung 5 zeigt die Komponenten der OSSOTIS-Architektur in einer Benutzt-Hierarchie. Die Pfeile stellen hierbei die Benutzt-Relationen dar, höher stehende Komponenten benutzen die tiefer stehenden Komponenten. Das Modell besteht aus fünf Schichten. Auf der Datenhaltungsschicht befindet sich die Hardware der Sensoren und Funktechniken sowie das Rohmaterial der Karten in Dateien. Die Datenverwaltungsschicht hat die Aufgabe, für diese drei Bereiche Adapter bereitzustellen, damit darüber liegende Schichten einheitlich darauf zugreifen können. Die kooperative Situationsanalyse (*KoSit-Komponente*) beinhaltet Elemente der Steuerungs- und der Anwendungsschicht, dies wird im nächsten Kapitel genauer spezifiziert. Auf der Präsentationsschicht befindet sich die Darstellung der Ergebnisse im Fahrzeug sowie ggf. die Interaktion mit dem Fahrer. Die optionale Navigationssoftware verteilt sich auf zwei Schichten, die gespeicherte Route ist Teil der Anwendungsschicht, die Logik der Routenberechnung ist Teil der Steuerungsschicht.

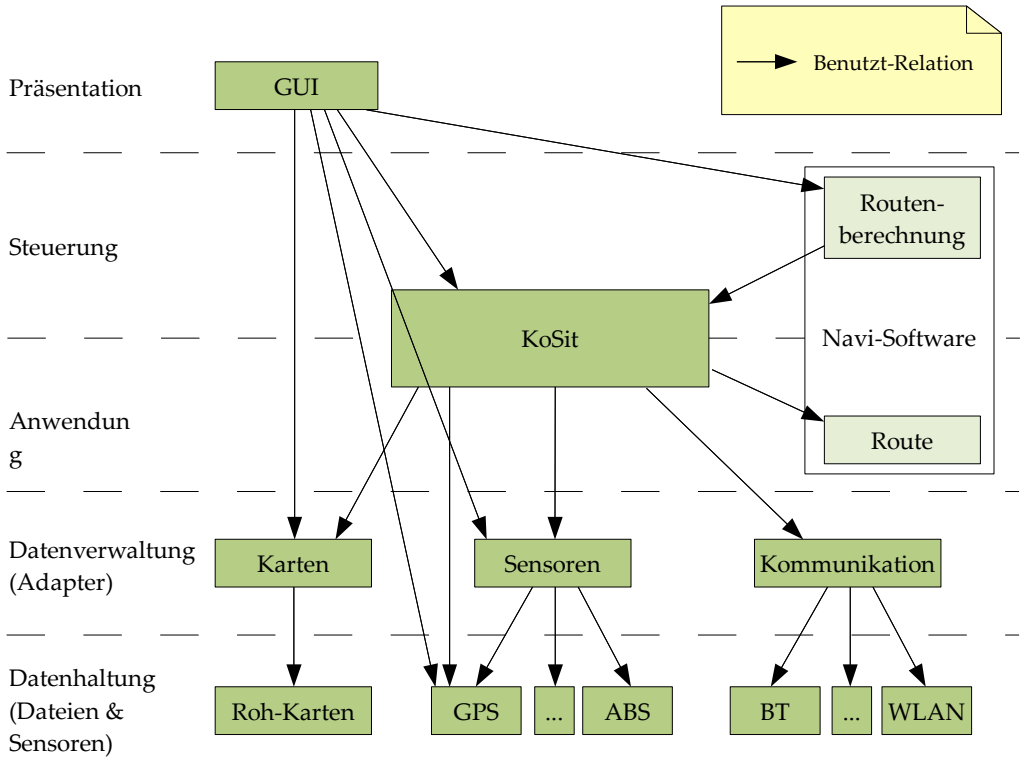

Abbildung 5: Benutzt-Hierarchie der OSSOTIS Architektur

3.2.1 GUI-Komponente

Die Benutzerschnittstelle dient hauptsächlich der Anzeige der Straßenkarten (bei vorhandener Navigationssoftware in Verbindung mit der geplanten Route), der durch die KoSit-Komponente bereitgestellten Verkehrsmeldungen sowie der Sensordaten und der GPS-Position. Außerdem können Eingaben entgegengenommen werden, um Parameter der Komponenten zu verändern oder die Navigationskomponente zu nutzen.

3.2.2 Navi-Software-Komponente

Die Navigationssoftware in Form der *Navi-Software-Komponente* ist nicht Teil des OSSOTIS-Projekts, kann aber in dieses integriert werden. In diesem Fall können die Ergebnisse der kooperativen Situationsanalyse in Form von Verkehrsinformationen für die Routenberechnung genutzt werden. Außerdem könnten einzelne Verfahren der kooperativen Si-

tuationsanalyse die gespeicherte Route in ihre Berechnungen mit einbeziehen, um beispielsweise die Relevanz von Verkehrsinformationen anders zu bewerten, wenn diese zwar nicht auf der aktuellen Straße, aber auf der geplanten Route liegen.

3.2.3 Karten-Komponente

Die *Karten-Komponente* stellt das Kartenmaterial aus Roh-Karten über eine einheitliche Schnittstelle zur Verfügung. Dies erfordert die Datenhaltung der Roh-Karten und die Bereitstellung von Funktionen für Berechnungen auf dem Kartenmaterial wie Entfernungen oder die Zuordnung von GPS-Positionen zu Straßen. Als Rohdaten werden die Karten der OpenStreetMap ([48]) genutzt.

3.2.4 Sensoren-Komponente

Die Ansteuerung der Hardware-Sensoren ist Aufgabe der *Sensoren-Komponente*. Dieser Adapter stellt die Daten von verschiedenen Fahrzeugsensoren über eine einheitliche Schnittstelle bereit. Die Sensoren werden dabei über den CAN-Bus o.ä. abgefragt, die Werte vereinheitlicht und an die übergeordnete Komponente zurückgegeben.

3.2.5 GPS-Komponente

Die *GPS-Komponente* dient der Positionsbestimmung. Sie stellt den anderen Komponenten die aktuelle GPS-Position zur Verfügung. Da die GPS-Position im Gegensatz zu anderen Sensorwerten sehr häufig und von mehreren Komponenten benötigt wird, hat dieser Sensor eine Sonderstellung und ist nicht nur über die *Sensoren-Komponente* erreichbar.

3.2.6 Kommunikations-Komponente

Die *Kommunikations-Komponente* dient der Nutzung des Ad-Hoc-Netzwerk zum Austausch von Fahrzeugdaten und Verkehrsmeldungen mit anderen Teilnehmern. Dieser Adapter kapselt die verschiedenen Übertragungstechniken, z.B. WiFi nach IEEE 802.11g/p oder Bluetooth. Die Komponente stellt Methoden zum Verschicken und Empfangen von Verkehrsmeldungen und Sensordaten zur Verfügung. Sie steuert den Medienzugriff und die Auslastung der Bandbreite im VANET.

3.2.7 Kooperative Situationsanalyse (KoSit)-Komponente

Die in der vorliegenden Arbeit behandelte kooperative Situationsanalyse ist in Form der *KoSit-Komponente* zentraler Bestandteil der OSSOTIS-Architektur. Sie benutzt die Straßenkarten, die Sensordaten und die Kommunikationskomponente um Fahrzeugdaten zu Verkehrsmeldungen zu veredeln. Innerhalb der Komponente besteht die Möglichkeit, unterschiedliche Verfahren einzubinden um verschiedene Muster (beispielsweise Stau, Wetter, Hindernisse) in den Rohdaten zu erkennen. Neben den Fahrzeug-lokal erstellten eigenen werden auch fremde Sensor-Rohdaten zur Erkennung von Verkehrssituationen genutzt, dies geschieht mit Hilfe der Kommunikationskomponente.

KoSit speichert und verwaltet die Fahrzeugdaten und Verkehrsmeldungen für spätere Analysen und Übertragungen an andere Fahrzeuge und entscheidet, welche Fahrzeugdaten und Verkehrsmeldungen mittels der Kommunikationskomponente übertragen werden sollen.

4 Spezifikation der KoSit-Komponente

4.1 Überblick

Die Infrastruktur für die kooperative Situationsanalyse umfasst zwei Aspekte: zuerst wird auf die KoSit-Komponente als Teil der OSSOTIS-Architektur eingegangen. Anschließend wird eine davon losgelöste Simulationsumgebung zum Testen von Verfahren zur kooperativen Situationsanalyse konzipiert. Die Simulationsumgebung verwendet Testreihen mit möglichst realitätsnahen Verkehrsdaten, um die Effizienz und Effektivität verschiedener Verfahren unabhängig von der OSSOTIS-Architektur vorab bestimmen zu können.

Zur begrifflichen Unterscheidung der Rohdaten aus den Sensoren von generierten Verkehrsinformationen werden folgende Begriffe definiert:

Sensordaten sind aus Fahrzeugsensoren ausgelesene Rohdaten des eigenen oder eines fremden Fahrzeugs.

Verkehrsinformation sind das Ergebnis der kooperativen Situationsanalyse, also aus Sensordaten abstrahierte Beschreibungen einer Verkehrssituation.

Verkehrsdaten sind eine Menge aus Sensordaten und Verkehrsinformationen.

Relevante Verkehrsdaten sind die Typen der Verkehrsdaten, die ein Verfahren in seinen Algorithmus einbezieht, an Hand derer es also Verkehrssituationen erkennt. Bei Erzeugung relevanter Verkehrsdaten durch fahrzeuglokale Sensoren bzw. Empfang dieser aus dem VANET soll das jeweilige Verfahren ausgeführt werden.

4.2 Anforderungen an die KoSit-Komponente

Das zu entwickelnde System soll die Aufgabe der kooperativen Situationsanalyse (KoSit-Komponente) in der parallel zu dieser Arbeit entworfenen OSSOTIS-Architektur von Christopher Schaumann ([47]) übernehmen. Da noch nicht absehbar ist, welche Aufgaben die einzelnen Komponenten genau erfüllen und wie die Kommunikation untereinander spezifiziert wird, verwende ich im Folgenden das in Abschnitt 3.2 vorgestellte, vorläufige Schichtenmodell der OSSOTIS-Architektur.

Die notwendige Integration in die infolgedessen vorgegebene Architektur stellt mehrere Anforderungen an das System. Daneben ergeben sich eine Reihe weiterer Anforderungen durch die Zielsetzung dieser Arbeit und die in Abschnitt 2.4 vorgestellten, bereits existierenden SOTIS-ähnlichen Architekturen.

Funktionale Anforderungen

- **Einbindung in Schichtenmodell:** Die OSSOTIS-Architektur ist ein schichtenbasiertes System. Die Abhängigkeitsbeziehungen der KoSit-Komponente müssen eingehalten werden, insbesondere dürfen nur hierarchisch tiefer liegende Komponenten benutzt werden.
- **Bereitstellung von Schnittstellen:** Die Ergebnisse der KoSit-Komponente werden von anderen Komponenten des Systems benutzt, folglich müssen die dafür notwendigen Schnittstellen bereitgestellt werden.
- **Unterstützung verschiedener Verfahren:** Das zu entwickelnde System soll die Infrastruktur für unterschiedliche Verfahren der kooperativen Situationsanalyse bereitstellen. Infolgedessen müssen sich mehrere Verfahren einbinden lassen, die mit den Rohdaten arbeiten und Verkehrsinformationen generieren.
- **Speicherung und Übertragung der Sensordaten/Verkehrsinformationen:** Aus der Zielsetzung des OSSOTIS-Projekts bzw. der Funktionsweise kooperativer Situationsanalysen ergibt sich, dass Sensor-Rohdaten und generierte Verkehrsinformationen zwischen den Fahrzeugen ausgetauscht werden müssen. Da die Struktur des zu Grunde liegenden Fahrzeug-Ad-hoc-Netzwerkes im fließenden Verkehr dynamisch ist und die Daten auch über die Funkreichweite hinaus verbreitet werden sollen, müssen die vorliegenden Daten zur späteren Kommunikation gespeichert werden.

Nicht-funktionale Anforderungen

- **Flexibilität bzgl. der Gesamtarchitektur:** Da die OSSOTIS-Architektur und damit die Schnittstellen zwischen den Komponenten noch nicht endgültig spezifiziert ist, soll die KoSit-Komponente möglichst einfach an eine veränderte Architektur angepasst werden können.
- **Nutzung der Ausführungsumgebung:** Die OSSOTIS-Architektur nutzt voraussichtlich eine OSGi-Ausführungsumgebung. Die KoSit-Komponente sollte innerhalb dieser ausgeführt werden. Dies setzt eine Java-basierte Implementierung voraus.
- **Nutzung von Open Source Technologien:** Mit dem OSSOTIS-Projekt als Open Source System sollen die Probleme proprietärer Lösungen umgangen und ein offener Standard forciert werden. Damit muss auch die KoSit-Komponente ausschließlich auf quelloffene Technologien setzen.
- **Skalierbarkeit:** Ein typisches Merkmal von Fahrzeugtechnik (vgl. Kapitel 2.2 und 2.3) ist ein unterschiedlich hoher Verbreitungsgrad. Während der Einführungsphase sind nur wenige Fahrzeuge mit der Technik ausgestattet, bei breiter Akzeptanz steigt der Anteil potentiell bis zur Marktsättigung[16]. Lars Wischof et al. haben nachgewiesen[17], dass SOTIS auch bei einem geringen Anteil von weniger als drei Prozent am Fahrzeugaufkommen funktionsfähig sein können. Das System sollte sowohl mit einer niedrigen als auch mit einer hohen Fahrzeugabdeckung brauchbare Ergebnisse liefern.
- **Fehlertoleranz:** Da das OSSOTIS ein quelloffenes, dezentrales System ist, können beispielsweise Übertragungsfehler oder Inkompatibilitäten durch fahrzeugübergreifend unterschiedliche Software-Versionen auftreten. Außerdem könnte versucht werden, das System durch Versenden manipulierter Pakete zu stören. Die KoSit-Komponente sollte bei einer geringen Fehlerquote davon unabhängig funktionieren.
- **Unabhängigkeit von Sensordaten:** Die Menge der im Fahrzeug verfügbaren Sensoren wird stetig erweitert (vgl. Abschnitt 2.2). Die Infrastruktur für die kooperative Situationsanalyse sollte so ausgelegt sein, dass auch neue bzw. bisher unbekannte Sensordaten verarbeitet werden können. Hierfür soll lediglich die Einbindung eines neuen Verfahrens notwendig sein.

16 Beispielhaft ist der Gesamtbestand an Kfz-Navigationsgeräten in Europa innerhalb von sechs Jahren von ca. 10.000 (1995) auf 2,8 Mio. (2001) gestiegen. [50]

17 [49], S.237

- **Einfache Austauschbarkeit der Verfahren:** Einzelne Verfahren sollen möglichst einfach in das System eingebunden bzw. aus dem System entfernt werden können.
- **Unterstützung der Simulationsumgebung:** Die KoSit-Komponente sollte so konzipiert sein, dass sich die einzelnen Verfahren in der Simulationsumgebung unabhängig von der Gesamtarchitektur evaluieren lassen.

4.3 Architektur der KoSit-Komponente

In diesem Abschnitt wird die Funktionsweise und der Aufbau der KoSit-Komponente beschrieben. Aus den Anforderungen ergeben sich zusammengefasst drei Funktionsbereiche: die Bereitstellung und Nutzung der Schnittstellen zur Gesamtarchitektur, die Verwaltung der Sensordaten und generierten Verkehrsinformationen und die einzelnen Verfahren der kooperativen Situationsanalyse. Um die Anforderungen bezüglich der Austauschbarkeit der Verfahren zu erfüllen, ist die logische Trennung dieses Bereiches notwendig. Möglich ist also die Aufteilung in zwei bzw. drei Komponenten. Eine weitere Unterteilung auf der Ebene der Infrastruktur ist nicht sinnvoll, allerdings können die Verfahren abhängig von ihrer Komplexität weiter strukturiert werden. Ein wichtiger Vorteil der Aufteilung in drei statt zwei Komponenten ist die Kapselung der Verkehrsdaten. Dies erlaubt eine einfachere Änderung der Datenverwaltung, z.B. zur Nutzung einer anderen Datenbank, ohne Änderungen an der Schnittstelle zu den Verfahren. Diese weitere Modularisierung vereinfacht auch die spätere Anpassung an die endgültige OSSOTIS-Architektur, da nur ein Teilbereich der KoSit-Komponente verändert werden muss und der Rest unverändert weiter genutzt werden kann. Durch diese Abstraktionsebene und damit den Verzicht auf den direkten Zugriff auf die Verkehrsdaten ist die Verarbeitungsgeschwindigkeit dagegen potentiell niedriger.

Die Aufteilung erfolgt in die folgenden drei Komponenten: einen *KoSit-Kern* zur Bereitstellung und Nutzung der Schnittstellen zu den anderen Komponenten, einer Komponente zur Verwaltung der Verkehrsdaten (*Wissensdatenbank*) und einer Schnittstelle zur Kapselung der einzelnen Verfahren. Abbildung 6 zeigt die einzelnen Komponenten im Kontext der Gesamtarchitektur.

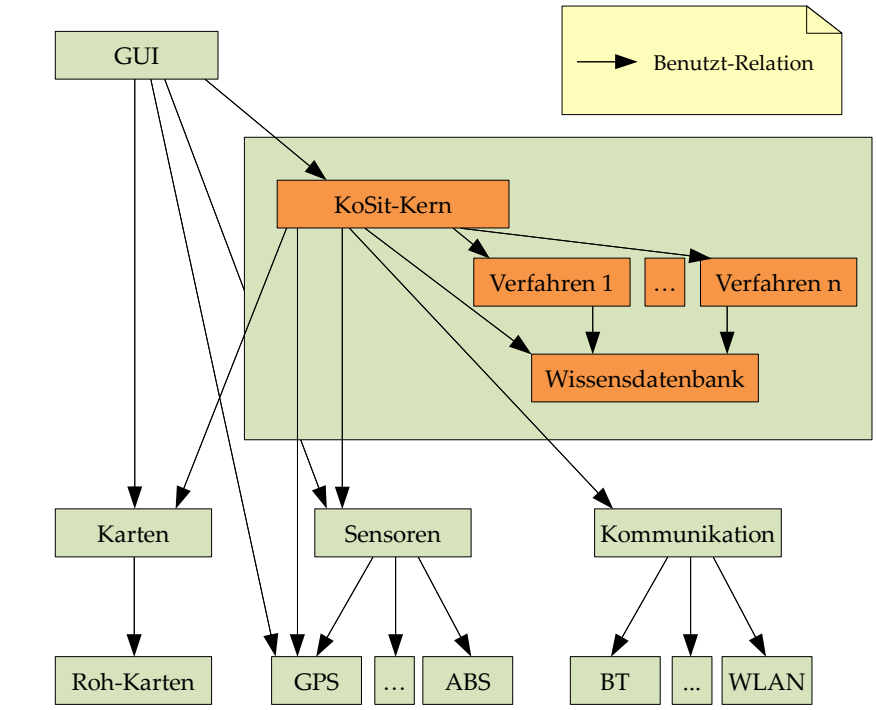

Abbildung 6: Aufbau KoSit-Komponente

Die Wissensdatenbank übernimmt die fahrzeuglokale Speicherung der eingegangen und im Fahrzeug erzeugten Sensordaten und Verkehrsinformationen in geeigneten Datenstrukturen, sie wird von den beiden anderen Modulen benutzt. Die einzelnen Verfahren werden vom KoSit-Kern aufgerufen und benutzen ausschließlich die Wissensdatenbank, um in den gespeicherten Sensordaten und Verkehrsmeldungen nach Mustern bestimmter Verkehrssituationen zu suchen.

Ablauf der kooperativen Situationsanalyse

Die kooperative Situationsanalyse im VANET verläuft in Anlehnung an die Traffic Information Distribution Technique von Lars Wischof et al. (siehe [22]) als dreistufige Technik: Empfangen, analysieren und senden von Verkehrsdaten (siehe Abbildung 7).

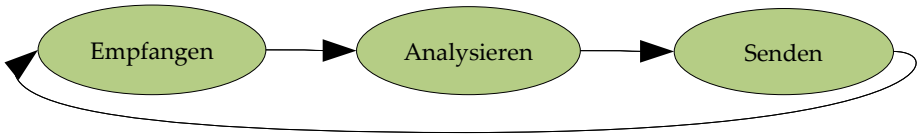

Abbildung 7: Verkehrsdatenfluss in der kooperativen Situationsanalyse

Im Gegensatz zu dieser werden im hier entwickelten System allerdings auch fremde Sensor-Rohdaten übertragen. Ohne diese Verbreitung der Sensordaten über die Funkreichweite einzelner Fahrzeuge hinaus wäre die Erkennung komplexerer Verkehrssituationen sehr schwierig, vor allem wenn nur wenige Fahrzeuge mit der Technik ausgestattet sind. In [22] wird nur auf die Erkennung des „Straßenzustands“ (in Bezug auf Stausituationen, ermittelt aus der Durchschnittsgeschwindigkeit) eingegangen, wofür die Übertragung generierter Verkehrsinformationen ausreichend ist.

Der KoSit-Kern bekommt von der Sensoren- und der Kommunikationskomponente generierte Sensordaten bzw. empfangene Sensordaten und Verkehrsinformationen. Diese übergibt er an die Wissensdatenbank, die sie, falls noch nicht vorhanden, speichert oder andernfalls den nochmaligen Empfang vermerkt. Anschließend prüft der KoSit-Kern für alle vorhandenen Verfahren, ob der Typ für sie relevant ist und ruft diese Verfahren gegebenenfalls auf. Dabei wird eine verfahrensspezifische und eine datentypspezifische Wartezeit zwischen zwei Aufrufen eingehalten, um in Situationen mit hoher Kommunikationsdichte und bei Sensoren mit häufigen, periodischen Werten das System nicht zu überlasten und den Verfahren ausreichend Zeit zur Bearbeitung zu geben. Im Gegensatz zum periodischen Starten aller Verfahren hat dies den Vorteil, schnell auf Verkehrsdaten reagieren und passende Verkehrsinformationen generieren zu können. Anschließend geben die Verfahren ihre generierten Verkehrsinformationen zurück, die der KoSit-Kern an die Wissensdatenbank zur Speicherung weiterleitet. Der KoSit-Kern leitet diese außerdem an darüber liegende Komponenten weiter und verbreitet sie im Rahmen der periodischen Übertragung an die Kommunikationskomponente im Ad-Hoc-Netz.

Nachfolgend ist der beschriebene Ablauf der kooperativen Situationsanalyse in einem Aktivitätsdiagramm[18] dargestellt (Abbildung 8). Die Details der Schnittstellen zu den beteiligten Komponenten der Gesamtarchitektur sind in Abschnitt 4.5 beschrieben.

18 In UML2-Notation nach [51]

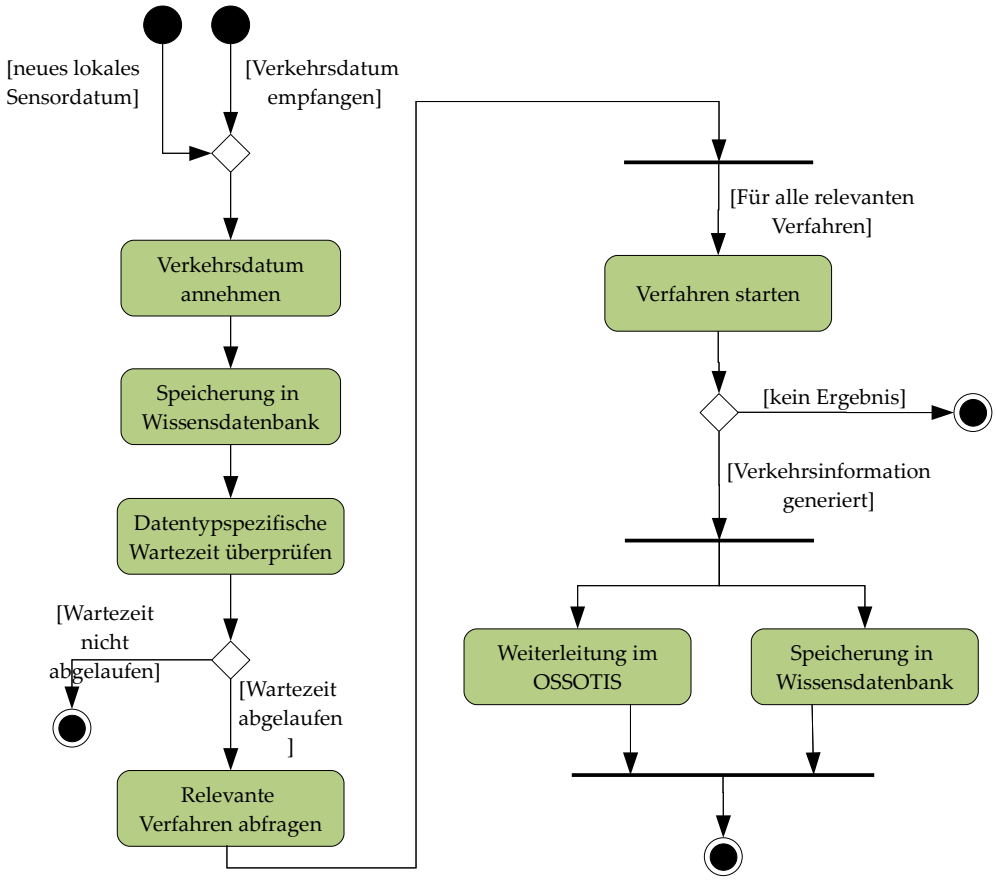

Abbildung 8: Aktivitätsdiagramm Kooperative Situationsanalyse

4.3.1 Spezifikation KoSit-Kern

Der KoSit-Kern steuert die kooperative Situationsanalyse und den Datenfluss innerhalb der KoSit-Komponente. Er implementiert die in Abschnitt 4.5 beschriebenen Schnittstellen zur Gesamtarchitektur und kontrolliert die fahrzeugübergreifende Kommunikation.

Über die Kommunikationskomponente werden eigene und von anderen Fahrzeugen empfangene Rohdaten und selbst generierte Verkehrsinformationen verschickt. Fremde Verkehrsinformationen werden nicht direkt weitergeleitet, sondern dienen als Indiz für eigene Verkehrsinformationen. Falls noch weitere, die Situation betreffende Verkehrsdaten vorliegen, kann auch eine gegenteilige Verkehrsinformation generiert werden, um die (auch bei anderen Fahrzeugen) vorliegende Information zu revidieren. An übergeordnete Komponenten werden also nur selbst generierte Meldungen weitergeleitet.

Steuerung der Kommunikation

Der Austausch von Verkehrsdaten mit anderen Fahrzeugen ist ein wesentlicher Bestandteil der kooperativen Situationsanalyse, der von der KoSit-Komponente und der Kommunikationskomponente ausgeführt wird. Innerhalb des KoSit-Kerns erfolgt die Entscheidung, welche Verkehrsdaten periodisch per Broadcast im VANET übertragen werden sollen. Diese werden priorisiert, damit wichtige Daten zuerst verschickt bzw. weniger wichtigen Daten vorgezogen werden, falls die Bandbreite für eine vollständige Übertragung nicht ausreicht. Die Bandbreite der verwendeten Funktechnik und die Wartezeit zwischen zwei Übertragungen müssen in der KoSit-Komponente eingestellt werden. Die weiteren Aspekte der Kommunikation wie der Medienzugriff und die Kanalaufteilung übernimmt die Kommunikationskomponente und ist nicht Teil dieser Arbeit. Effiziente Verfahren für die Kommunikation in VANETs wurden bereits erforscht[19].

Die Priorität der Verkehrsdaten wird fahrzeuglokal berechnet und soll den Nutzen der Daten für die anderen Teilnehmer des VANETs widerspiegeln. Dieser ist vor allem abhängig von der Bekanntheit der Information im VANET, außerdem sollen generierte Verkehrsinformationen gegenüber einfachen Sensordaten vorgezogen werden.

Stephan Eichler et al. nutzen in [52] ähnliche Daten, um für jede Verkehrsinformation einen individuellen Nutzen („*MessageBenefit*“) zu berechnen. Dazu werden *N* Einzelwerte des Nachrichtenkontextes (*Message Context m*, z.B. Alter der Nachricht, letzte Übermittlung im VANET), des Fahrzeugkontextes (*Vehicle context v*, z.B. Informationen zur Geschwindigkeit oder zur Straße) und des allgemeinen Kontextes (*Information context i*, z.B. Tageszeit, Abstand zum Reisestartpunkt) mit Hilfe einer applikationsabhängigen Funktion *b* zu einem Wert zwischen 0 (kein Nutzen) und 1 (hoher Nutzen) aggregiert, Faktor *a* gewichtet die einzelnen Werte:

$$MessageBenefit = \frac{1}{\sum_{i=1}^{N} a_i} * \sum_{i=1}^{N} a_i * b_i(m, v, i) \quad [20]$$

Weitergehende Informationen zum Algorithmus finden sich in der in oben genannten Literatur. Auch im NOW-Projekt wurde eine ähnliche, relevanzbasierte Kommunikation verwendet (siehe Abschnitt 2.4.7). Die Berechnung des MessageBenefit zielte auf ein System zur Verbreitung von Informationen über lokale Gefahrenquellen ab, bei dem Verkehrsdaten nur bei vorliegenden Gefahren und nicht kontinuierlich kommuniziert wer-

19 z.B. in [49], S. 238 oder in [52]

20 [52], S. 4f

den. Gegenüber den Anwendungen des OSSOTIS ist die Menge zu übertragender Informationen infolgedessen deutlich kleiner, insbesondere durch den Verzicht auf fahrzeugübergreifende Sensordaten.

Zur Berechnung der Prioritäten der Verkehrsdaten innerhalb der KoSit-Komponente soll deshalb eine vereinfachte Formel verwendet werden. Folgende Parameter stehen zusammengefasst zur Berechnung zur Verfügung:

- Anzahl, wie oft das Verkehrsdatum empfangen wurde (*Anzahl n*). Dies spiegelt die Bekanntheit des Datums im VANET wieder.
- Datum und Uhrzeit des letzten Empfangs des Verkehrsdatums, als Alter in Sekunden (*letzter Empfang e*).
- Das Alter des Verkehrsdatums in Sekunden (*Alter a*). Aktuelle Verkehrsdaten sind potentiell wichtiger und die Wahrscheinlichkeit, dass das Datum beim Empfänger noch nicht bekannt ist, ist höher.
- Eigenschaft, ob es sich um ein lokales beziehungsweise fremdes Verkehrsdatum handelt (*lokales Verkehrsdatum l: 0/1*). Der Ursprung lässt sich auch bei empfangenen Daten feststellen, da lokal erzeugte Daten in der Wissensdatenbank gespeichert und entsprechend markiert sind.
- Eigenschaft, ob es sich um ein Sensordatum oder eine Verkehrsinformation handelt (*Verkehrsinformation v: 0/1*).
- Eigenschaft, ob das Verkehrsdatum die aktuelle Straße betrifft (*Straße s: 0/1*).

Entgegen der Berechnung des MessageBenefit sollen nicht alle dieser Parameter bei steigendem Wert den Nutzen bzw. die Priorität des Verkehrsdatums erhöhen. Die Parameter *Anzahl n*, *letzter Empfang e* und *Alter a* verringern die Priorität, je höher ihr Wert ist.

In Anlehnung an die Formel des MessageBenefit wird die Priorität nach der unten stehenden Formel berechnet, die die oben genannten Anforderungen erfüllt:

$$Priorität = \frac{\sum_{i=1}^{3} G_i * M_i + \frac{\log 5}{6 * \log(a+2)}}{G_4 * M_4 + 1}$$

$$M = (s, l, v, n), \quad G = \left(\frac{1}{8}, \frac{1}{4}, 1, \frac{1}{100}\right)$$

Die Parameter *n, l, s* und *v* werden gewichtet summiert, das Alter *a* geht logarithmisch in die Berechnung ein – ein Alter von wenigen Sekunden erhöht die Priorität bedeutend, mit steigendem Alter wird der Wert dieses Parameters kleiner. Parameter *a* (*Anzahl*) ist ohne weitere Einschränkungen nicht hilfreich, da sich das VANET schnell verändert und alte Broadcast-Informationen nichts über die aktuelle Informationsverteilung aussagen. Deshalb werden mit dem Parameter *n* nur jeweils die Verkehrsdaten gezählt, die innerhalb der letzten 60 Sekunden empfangen wurden. Höhere Werte für *n* verringern die Priorität.

Die Formel liefert für die Priorität Werte zwischen 0 und 1 bei Sensordaten bzw. zwischen 0 und 2 bei Verkehrsinformationen. Die Priorität für Verkehrsinformationen fällt erst bei großem *n* unter 1, so dass diese in der Regel den Sensordaten vorgezogen werden. Zur Verdeutlichung sind in Tabelle 2 einige Beispielwerte aufgeführt:

Beschreibung	s	l	v	n	a	Priorität
Neue, lokal generierte Verkehrsinformation	1	1	1	0	0	1,76
selten empfangene, lokale, ältere Verkehrsinformation	1	1	1	2	180	1,40
Selten empfangene, fremde, aktuelle Verkehrsinformation	1	0	1	2	30	1,18
Häufig empfangene, fremde, aktuelle Verkehrsinformation	1	0	1	40	30	0,86
Sehr häufig empfangene, fremde, ältere Verkehrsinformation; andere Straße	0	0	1	100	180	0,53
Neues, lokales Sensordatum	1	1	0	0	0	0,76
Fremdes, älteres Sensordatum; gleiche Straße	1	0	0	5	180	0,17
Fremdes, älteres Sensordatum; andere Straße	0	0	0	5	180	0,05

Tabelle 2: Beispielwerte Prioritäten der Verkehrsdaten

Die Priorität wird direkt beim periodischen Abruf der vorliegenden Verkehrsdaten aus der Wissensdatenbank zur Weiterleitung an die Kommunikationskomponente berechnet. Je nach Auslastung des Netzwerkes können nur Verkehrsdaten oberhalb einer bestimmten Priorität versendet werden.

Steuerung der Verfahren

Eine weitere Aufgabe des KoSit-Kerns ist es, die eingebundenen Verfahren zur kooperativen Situationsanalyse zu steuern. Dies umfasst die Initialisierung der Verfahren sowie deren Aufruf, wenn neue relevante Daten vorliegen. Die Initialisierung erfolgt beim Start der KoSit-Komponente und erneut auf Anforderung, z.B. wenn der Benutzer weitere Verfahren einbinden möchte. Hierbei werden alle in einem definierten Plugin-Verzeichnis verfügbaren Verfahren registriert und abgefragt, welche Verkehrsdaten für ihren Algorithmus relevant sind und welchen Typ von Verkehrsinformation sie generieren.

Beim Eingang neuer Verkehrsdaten prüft der KoSit-Kern an Hand der bei der Initialisierung generierten Liste, welche Verfahren bei dem vorliegenden Sensordaten- bzw. Verkehrsinformationstyp gestartet werden müssen. In der Liste ist außerdem eine verfahrensspezifische Wartezeit pro Verkehrsdatum eingetragen, die zwischen zwei Aufrufen des Verfahrens vergangen sein muss. Damit wird sichergestellt, dass das Verfahren ausreichend Zeit zur Ausführung bekommt und das System bei einer hohen Dichte an Kommunikationspartnern nicht durch zu häufige Verfahrensdurchläufe überlastet wird. Die Wartezeiten werden von den Verfahren bei der Initialisierung bereitgestellt (weitere Details dazu in Abschnitt 4.3.3). Der KoSit-Kern muss demzufolge nachhalten, wann welches Verfahren wegen welchem neuen Verkehrsdatentyp zuletzt gestartet wurde.

Das aufgerufene Verfahren gibt gegebenenfalls die generierten Verkehrsinformationen zurück. Diese leitet der KoSit-Kern an übergeordnete Komponenten weiter und speichert sie in der Wissensdatenbank, wo sie zur Übertragung im VANET bereitstehen.

4.3.2 Spezifikation Wissensdatenbank

Die Wissensdatenbank stellt den Verfahren und dem KoSit-Kern Methoden zur Speicherung und zum Abruf von Verkehrsdaten bereit. Dadurch wird die interne Datenverwaltung gekapselt, die anderen Komponenten der KoSit-Komponente benötigen keine Kenntnis über die implementierte Datenverwaltung.

Die Verfahren stellen folgende Anforderungen an die Wissensdatenbank:

- **Filterung:** Um die Muster von Verkehrsinformationen in den Verkehrsdaten zu erkennen, benötigen die Verfahren jeweils die für sie relevanten Teilmengen der verfügbaren Verkehrsdaten. Um die Laufzeit der Mustererkennung zu minimieren, sollte die abzurufenden Verkehrsdaten bereits vorher gefiltert werden können. Neben der Laufzeit verringert dies auch den Speicherbedarf. Beispielsweise benötigt ein Verfahren zur Stauerkennung in der Regel keine Verkehrsdaten bezüglich des

Wetters. Die Filterung sollte nach Alter, Erzeugung in fremdem/lokalem Fahrzeug sowie Typ des Verkehrsdatums möglich sein, wobei auch eine Liste möglicher Typen übergeben werden kann.

- **Geschwindigkeit:** Die Wissensdatenbank muss die angeforderten Verkehrsdaten möglichst schnell herausfiltern und zurückgeben, damit mehrere, parallel ausgeführte Verfahren sich nicht blockieren. Dies erfordert eine auf das notwendige beschränkte Wissensdatenbank, die Erkennungslogik wird in den Verfahren implementiert.

Der KoSit-Kern nutzt die Wissensdatenbank auf eine ähnliche Weise. Zur Berechnung der Prioritäten und darauf folgende Weiterleitung ins VANET ist der Abruf aller gespeicherten Verkehrsdaten notwendig, wobei diese nach dem Alter gefiltert werden können. Außerdem benötigt er eine Möglichkeit, neue Verkehrsinformationen und Sensordaten zu speichern.

Bereitgestellte Methoden

Um die oben genannten Anforderungen beider Komponenten zu erfüllen, werden folgende Methoden implementiert:

- `get(Filter f): Verkehrsdaten[] v` [21]
  Mit dieser Methode können Verkehrsdaten abgerufen werden. Das Filter-Objekt definiert die Einschränkungen (Alter, fremd/lokal, Sensordaten/Verkehrsinformationen, Typen) der zurückzugebenden Verkehrsdaten.
- `put(Verkehrsdatum v, Boolean lokal)`
  Mit dieser Methode können einzelne Verkehrsdaten gespeichert werden, übergeben wird das zu speichernde Verkehrsdatum und ob es sich um ein lokal erstelltes Verkehrsdatum handelt..

Weitere Details zu den als Parameter verwendeten Objekten werden in Abschnitt 6.3.1 erläutert.

21 Syntax: `Methodenname(Parameter): Rückgabe`

Datenverwaltung

Für die Verwaltung der Daten innerhalb der Wissensdatenbank stehen verschiedene Datenbankverwaltungssysteme (engl. *database management system*, abgekürzt *DBMS*) zur Auswahl. Durch die Speicherung eigener und fremder Sensor-Rohdaten wird die zu verwaltende Datenmenge schnell sehr umfangreich, so dass für die Filterung der Daten effiziente Suchalgorithmen verfügbar sein müssen. In eingebetteten Systemen, für die das System ausgelegt sein wird, ist nur wenig Arbeitsspeicher vorhanden. Infolgedessen müssen das Datenverwaltungssystem auch dann effizient arbeiten, wenn nicht der gesamte Datenbestand im Arbeitsspeicher vorliegt. Aktuelle DBMS lassen sich nach den verwendeten Datenmodellen in drei Klassen einteilen: relationale, objektorientierte und XML-basierte DBMS, außerdem existieren DBMS, die zwei Modelle kombiniert oder ein erweitertes Modell einsetzen. „Ältere Datenmodelle (die sogenannten satzorientierten Modelle, zu denen das Netzwerkmodell und das hierarchische Modell zählen) [werden hier nicht betrachtet], da diese Datenbanksysteme in absehbarer Zeit wohl nur noch historische Bedeutung haben werden." ([53]). In Tabelle 3 findet sich ein Vergleich der drei genannten Datenmodelle an Hand der für diese Arbeit relevanten Kriterien.

	relational	**objektorientiert**	**XML-basiert**
Abfragesprache	SQL	OQL	XPath, XQuery
Daten-strukturierung	Speicherung in flache Tabellen (Relationen)	Zusammenfassung in Objekttypen ähnlich zur objektorientierten Programmierung	Hierarchischer Aufbau, Strukturierung als Baum
Vorteile	Verbreiteter Standard, hohe Performanz, einfache und flexible Nutzung	Direkte Speicherung auch von komplexen Objekten der Anwendung in der Datenbank	Einfache Nutzung in Webservice-basierten Umgebungen
Nachteile	Umsetzung der Objekte aus der Anwendung auf Relationen nötig, komplexe Objektbeziehungen schwer abzubilden	Hoher Speicherbedarf durch Redundanzen, geringe Performanz bei großen Datenmengen	Geringe Performanz

Tabelle 3: Vergleich Datenbankverwaltungssysteme[22]

22 Siehe [53]

XML-basierte Datenbanken scheiden auf Grund der großen Datenmengen und der dafür benötigten Geschwindigkeit aus. Ein objektorientiertes Datenmodell hätte durch die Implementierung des Systems in Java den Vorteil, Objektinstanzen der Verkehrsdaten direkt in der Datenbank nutzen zu können. Allerdings sind die Verkehrsdaten einfach strukturiert und haben keine Beziehungen untereinander. Folglich wird für die Datenverwaltung eine relationale Datenbank genutzt. Die Modellierung des Datenbankschemas erfolgt in Abschnitt 6.3.1. Weitergehende Informationen über DBMS beziehungsweise Datenmodelle sind nicht Teil dieser Arbeit, hierfür verweise ich auf [53].

4.3.3 Schnittstelle zu den Verfahren

Die Einbindung der Verfahren in die KoSit-Komponente erfolgt über die Spezifikation einer Schnittstelle, die die Verfahren implementieren. Die Schnittstelle ist möglichst einfach gehalten, die Verfahren haben keine Beziehung zu Komponenten außerhalb der KoSit-Komponente. Damit ist die Entwicklung der Verfahren unabhängig von der Gesamtarchitektur möglich. Infolgedessen übertragen die Verfahren die von ihnen generierten Verkehrsinformationen nicht direkt im VANET, sondern geben sie als Rückgabe der Aufruf-Methode an die *KoSit-Kern*-Komponente zurück. Die Verfahren müssen dazu folgende Methoden bereitstellen:

Initialisierungs-Methode

Dieser Konstruktor wird einmalig zur Initialisierung des Verfahrens vom KoSit-Kern aufgerufen. Dieser übergibt eine Referenz zur Wissensdatenbank, auf der das Verfahren arbeiten soll.

Methode zum Abruf der Verkehrsdatenliste

Zurückgegeben wird ein Objekt, dass die für das Verfahren relevanten Verkehrsdaten sowie die Typen der von dem Verfahren generierten Verkehrsinformationen enthält. Für die relevanten Verkehrsdaten sind außerdem die Wartezeiten in Sekunden enthalten. Nachfolgend steht die Methodensignatur in Pseudocode.

```
getVerkehrsdatenliste(): Verkehrsdatenliste vdListe;
```

Start-Methode

Innerhalb der *start*-Methode erfolgt die Mustererkennung im Rahmen der kooperative Situationsanalyse. Beim Aufruf wird der Algorithmus auf der Wissensdatenbank ausgeführt, zurückgegeben wird eine Liste der erkannten Verkehrsinformationen bzw. eine leere Liste, falls keine bekannten Muster von Verkehrsinformationen gefunden wurden. Zur Unterstützung des Verfahrens wird ein aktueller Ausschnitt der Straßenkarte übergeben, die der KoSit-Kern von der Karten-Komponente abruft. Das Kartenmaterial liegt im OpenStreetMap-Format in XML vor (vgl. Abschnitt 3.2.3). Nachfolgend steht die Methodensignatur in Pseudocode.

```
start(Kartenmaterial karte): Verkehrsinformation[] v
```

4.4 Spezifikation der Verkehrsdaten

In den vorherigen Abschnitten wurde eine informelle Definition der Sensordaten und Verkehrsinformationen benutzt. Diese wird jetzt genauer spezifiziert und damit festgelegt, welche Datenfelder enthalten sein müssen.

4.4.1 Datenstruktur Verkehrsinformationen

Im IVHW-Projekt wurde eine Datenstruktur zur Kommunikation der Warnungen spezifiziert. Mit dieser können die elf möglichen Warnmeldungen[23] einheitlich übertragen werden. Die Gesamtgröße einer Meldung wurde auf 445 Bit begrenzt, um die Anforderungen an die Funktechnik zu verringern. Infolgedessen ergeben sich bei den einzelnen Datenfeldern Einschränkungen bei der Genauigkeit. Folgende Datenfelder sind vorgesehen ([18]):

- *Preamble, Start, Header*
- *Message ID*: Zufälliger Identifikationsschlüssel zur Wiedererkennung von Meldungen, wenn diese mehrfach übertragen werden. Die zur Verfügung stehenden 9 Bit erlauben nur die Unterscheidung zwischen 512 verschiedenen Meldungen.
- *Road Type:* Ermöglicht die Unterscheidung der betroffenen Straße in Autobahn, Landstraße, Stadtstraße und unbekannt.
- *Road ID:* Identifikationsschlüssel der betroffenen Straße.
- *Hazard Type:* Typ der Warnmeldung.

23 Für eine genaue Beschreibung der einzelnen Typen siehe [18], S. 2 ff.

- *Current Speed:* Aktuelle Geschwindigkeit des Fahrzeugs.
- *Position and trace data:* GPS-Position der Warnmeldung.
- *Error correction:* Prüfsumme zur Erkennung von Übertragungsfehlern.

André Ebner et. al. nutzen in ihrer „Prototype Implementation“ eines SOTIS in [54] ein ähnliches Datenpaket für den Austausch von Verkehrsinformationen (*Travel and Traffic Information*, kurz *TTI* genannt), das neben einem *SOTIS Header* mit allgemeinen, kommunikationsbezogenen Daten folgende Felder enthält (vgl. Abbildung 9):

- *Road ID:* Eindeutige Kennung der Straße.
- *Start/End Segment ID:* Der von der TTI betroffener Straßenabschnitt.
- *TTI Value:* Der Inhalt der Verkehrsinformation.
- *Time Stamp:* Zeitstempel der Erzeugung des *TTI Value.*

  (mehrfaches Auftreten von *TTI Value* und *Time Stamp* ist möglich)

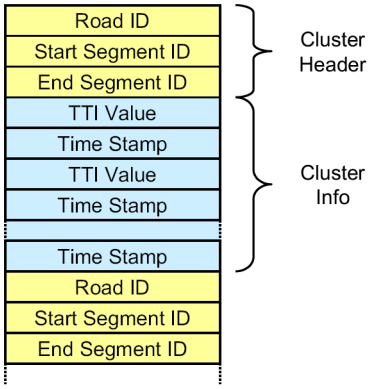

Abbildung 9: SOTIS Datenpaket [54]

Diese Informationen sind auch für die hier entwickelte kooperative Situationsanalyse notwendig. Zusätzlich wird ein Identifikationsschlüssel eingeführt, über den man den mehrfachen Eingang der gleichen Verkehrsinformation vom Eingang verschiedener Datenpakete zur gleichen Verkehrssituation (die also in der Regel von verschiedenen Fahrzeugen generiert wurden) unterscheiden kann. Die Einführung einer fahrzeugbezogenen Identifikation erlaubte auch die Zuordnung von von einem Fahrzeug mehrfach generierten Verkehrsinformationen mit gleichem Inhalt. Dies ist aus Gründen des Datenschutzes problematisch und könnte die Akzeptanz des SOTIS beeinträchtigen, diese Möglichkeit

wird daher nicht weiter betrachtet. Der Datenschutz in einem SOTIS bzw. einem VANET ist nicht Teil dieser Arbeit, dieses Thema wurde unter anderem von Matthias Gerlach ([55], [56]), Falko Dressler et. al. ([57]) und Krishna Sampigethaya et. al. ([58]) bearbeitet.

Die Position wird über zwei GPS-Punkte mit jeweils einer x- und einer y-Koordinate definiert. Damit kann die Verkehrssituation präziser positioniert werden als bei einer Segmentierung der Straßen. In der SOTIS-Implementierung des FleetNet Projekts wird eine Segmentgröße von 100 Metern genutzt[24], das ist für Verkehrssituationen mit kleiner Ausdehnung, deren Position präzise bestimmbar sein muss, beispielsweise Ölflecken oder Hindernisse, nicht praktikabel. Auch bei Stausituationen kann eine Abstraktion auf diese Abschnittsgröße negative Auswirkungen haben, wenn beispielsweise die Routenplanung Abfahrten oder Kreuzungen fälschlicherweise nicht in die Berechnung einbezieht.

Der Typ der Verkehrsinformation kennzeichnet die vorliegende Verkehrssituation eindeutig. Die möglichen Werte für den Typ der Verkehrsmeldung sind nicht statisch festgelegt, sondern können durch die verschiedenen Verfahren der kooperativen Situationsanalyse frei gesetzt werden. Dies ermöglicht die zukünftige Nutzung von neuen Typen, ohne die Architektur zu verändern. Mittels des Wertes *active* wird der Status der Verkehrsinformation festgelegt, also ob in den Verkehrsdaten festgestellt wurde, dass die benannte Verkehrssituation aktuell vorliegt (*active* ist 1) oder sich aufgelöst hat (*active* ist 0). Zusätzlich wird der Zeitpunkt der Erstellung der Verkehrsinformation festgehalten. Abbildung 10 fasst die Datenstruktur grafisch zusammen.

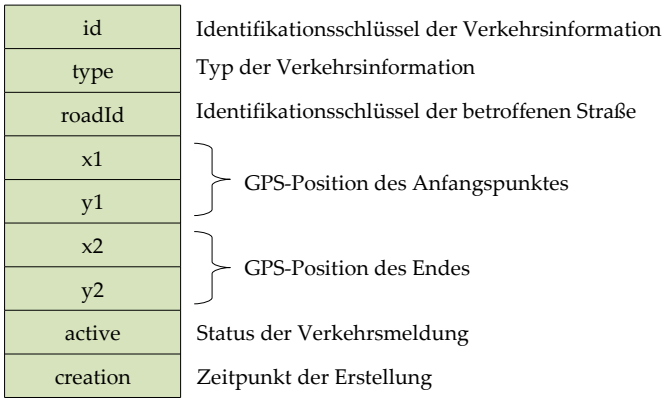

Abbildung 10: Datenstruktur Verkehrsinformation

24 Die Segmentgröße im SODAD-Verfahren kann in Abhängigkeit vom Straßentyp variieren. Vorgeschlagen werden 100 Meter für die aktuell befahrene Straße bzw. Autobahn, 200 Meter für Landstraßen in der näheren Umgebung. [49], S. 239

4.4.2 Datenstruktur Sensordaten

Für die Sensordaten sind weniger Datenfelder notwendig. Neben einem Identifikationsschlüssel des Sensortyps, *key*, sind diese der vom Sensor ausgelesene Wert, *value*, die zugehörige GPS-Position und der Zeitpunkt der Erstellung. Die betroffene Straße wird nicht gespeichert, da diese sich mittels der Karten-Komponente aus der GPS-Position bestimmen lässt. Diese Zuordnung kann gegebenenfalls im Verfahren zur kooperativen Situationsanalyse durchgeführt werden. Das Abfragen für alle Sensorwerte direkt bei der Erstellung würde das System stärker belasten[25].

Nachfolgend sind typische und für ein SOTIS relevante Fahrzeugsensoren mit dem jeweiligen Datentyp des Sensorwertes aufgelistet:

Fahrzeugsensor	Messwerte	Datentyp
ABS-Sensor	Radumfangsgeschwindigkeit	kontinuierlich
Bremse	ja/nein oder Intensität	binär oder Dezimalzahl
Frontradar	Abstand zum voraus fahrenden Fahrzeug	Dezimalzahl
Geschwindigkeitssensor	Aktuelle Geschwindigkeit	Dezimalzahl
Regensensor	Reflektierte Lichtmenge	Dezimalzahl

Tabelle 4: Fahrzeugsensoren und Datentypen der zugehörigen Werte[26]

Für die meisten Sensoren ist demnach eine periodisch ausgelesene, eindimensionale Dezimalzahl als Sensorwert ausreichend oder der Sensorwert lässt sich ohne relevanten Informationsverlust darauf reduzieren. Beim ABS-Sensor wird beispielsweise die kontinuierlich anliegende Wechselspannung als Radumfangsgeschwindigkeit interpretiert und ins Verhältnis zur Fahrzeuggeschwindigkeit gesetzt. Dies ist Aufgabe der Sensoren-Komponente beziehungsweise des jeweilige Sensor-Adapters. Die Beschränkung auf einen Dezimalwert reicht für die wichtigsten Anwendungsfälle aus und erleichtert den Umgang mit den Sensordaten im Vergleich zu komplexeren Objekten erheblich, vor allem in Bezug auf die kooperative Situationsanalyse, bei der in kurzer Zeit eine große Menge an Sensordaten betrachtet werden muss. Dies spiegelt sich in einem geringeren Speicherbe-

25 Es werden in der Regel deutlich mehr Sensorwerte erzeugt als in den Verfahren ausgewertet, da die Verfahren nur einzelne, ausgewählte Sensoren betrachten und zumindest vorerst davon ausgegangen werden kann, dass die Anzahl eingebundener Verfahren gering ist. Andernfalls ist zu überlegen, den Identifikationsschlüssel der betroffenen Straße in die Sensordaten zu integrieren, um eine mehrfache Abfrage der selben Koordinaten zu verhindern.

26 Beispielhafte Auswahl der für die kooperative Situationsanalyse wichtigsten Sensoren. Zur Sensortechnik siehe [5].

darf und effizienteren Algorithmen wieder. Messreihen o.ä. sind damit allerdings nur über mehrere einzelne Sensordaten abbildbar. Abbildung 11 zeigt die vollständige Datenstruktur eines Sensordatums.

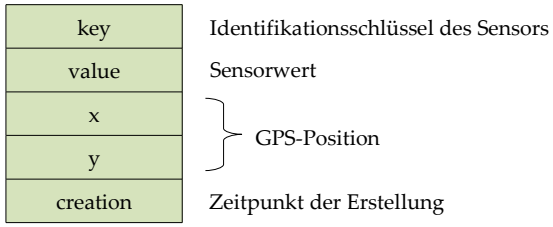

Abbildung 11: Datenstruktur Sensordatum

4.5 Integration in die OSSOTIS-Architektur

In den folgenden Abschnitten werden die Schnittstellen zu den anderen Komponenten der vorläufigen OSSOTIS-Architektur (siehe Abschnitt 3.2) beschrieben. Abbildung 12 verdeutlicht die Einbindung der KoSit-Komponente in das Gesamtsystem. Die Komponenten GUI und Routenberechnung stehen in der Benutzt-Hierarchie oberhalb der KoSit-Komponente und benutzen diese. Die KoSit-Komponente benötigt die Komponenten Karten, Sensoren, GPS und Kommunikation. Die Navi-Software mit den Komponenten Routenberechnung und Route ist optional und nicht Teil der OSSOTIS-Architektur, deshalb wird darauf im Folgenden nicht weiter eingegangen.

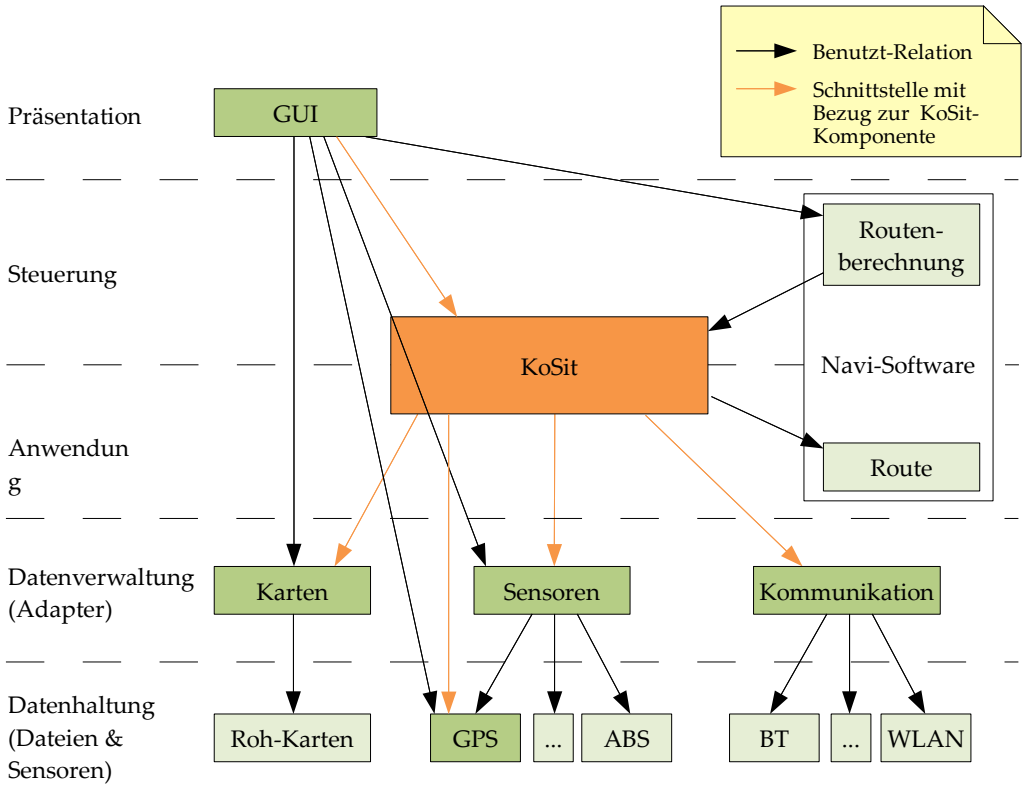

Abbildung 12: Schnittstellen der KoSit-Komponente

4.5.1 Schnittstelle zu GUI / Routenberechnung

Zur Nutzung der generierten Verkehrsmeldungen durch übergeordnete Komponenten stellt die KoSit-Komponente eine ereignisbasierte Schnittstelle *Verkehrsinformation* nach dem *Observer-Pattern*[27] bereit. An den Verkehrsmeldungen interessierte Komponenten können sich bei der KoSit-Komponente registrieren und erhalten daraufhin alle generierten Verkehrsmeldungen. Dazu gehören auch von anderen Fahrzeugen empfangene Verkehrsmeldungen, die die KoSit-Komponente bestätigt hat. Die KoSit-Komponente überträgt sieben Werte, die die Weiterverarbeitung der Verkehrsmeldung ermöglichen:

27 Siehe [59], S. 287 ff.

Bezeichnung	Beschreibung	Datentyp
type	Typ der Verkehrsmeldung	Zeichenkette
roadId	Identifikationsschlüssel der betroffenen Straße	Zeichenkette
x1	x-Koordinate des Anfangs der Verkehrssituation	Dezimalzahl
y1	y-Koordinate des Anfangs der Verkehrssituation	Dezimalzahl
x2	x-Koordinate des Endes der Verkehrssituation	Dezimalzahl
y2	y-Koordinate des Endes der Verkehrssituation	Dezimalzahl
active	Status der Verkehrsmeldung 1: gemeldete Verkehrssituation wurde erkannt 0: gemeldete Verkehrssituation besteht nicht mehr	binär

Tabelle 5: Rückgabeparameter der Schnittstelle Verkehrsinformation

Die übergeordneten Komponenten müssen auf das jeweilige Verfahren der KoSit-Komponente bezüglich der *type*-Werte aufeinander abgestimmt sein, für sie nicht relevante Verkehrsmeldungen können durch einen Zeichenkettenvergleich ausgefiltert werden. Die betroffene Straße wird durch die auf den Kartendaten basierende *roadId* bestimmt, die für jede Straße eindeutig ist. Die Koordinaten stellen Anfang und Ende der Verkehrssituation dar. Daraus lässt sich bei mehrspurigen Straßen die Fahrtrichtung ableiten, diese wird nicht gesondert übertragen. Um auch die Erkennung von aufgelösten Verkehrssituationen zuverlässig weiterverarbeiten zu können wird der boolesche Status *active* übergeben. Damit können Straßenbereiche als frei von der jeweiligen Verkehrssituation gekennzeichnet werden. Dies hat den Vorteil, dass Verkehrsmeldungen nicht von allen Komponenten gespeichert werden müssen, um widerrufbar zu sein.

Um die Schnittstelle einfacher zu gestalten, generiert die KoSit-Komponente bei Verkehrssituationen die mehrere, z.B. ineinander übergehende Straßen betreffen, hierfür mehrere Verkehrsmeldungen des gleichen Typs. Durch die Kartendaten lässt sich der Zusammenhang wiederherstellen, dies ist Aufgabe der übergeordneten Komponente.

4.5.2 Schnittstelle zu Karten

Für den Zugriff auf die Straßenkarten im Rohformat muss die Komponente Karten zwei Schnittstellen anbieten. Den Verfahren der kooperativen Situationsanalyse müssen Straßenkarten bzw. Ausschnitte davon bereitgestellt werden sowie die Zuordnung von einem GPS-Punkt zu einer Straße möglich sein. Erstere werden über die Schnittstelle *Kartenausschnitt* im XML-Format abgerufen. Dazu müssen vier Koordinaten angegeben werden,

die den rechteckigen Kartenausschnitt festlegen (Tabelle 6). Die Rückgabe enthält alle darin enthaltenen Straßen im OpenStreetMap-Format[28], dass Straßen als eine Folge von verbundenen Punkten und Kreuzungen über gemeinsame Punkte definiert.

Parameter	Beschreibung	Datentyp
x1	Erste GPS-x-Koordinate des Kartenausschnitts	Dezimalzahl
y1	Erste GPS-y-Koordinate des Kartenausschnitts	Dezimalzahl
x2	Zweite GPS-x-Koordinate des Kartenausschnitts	Dezimalzahl
x2	Zweite GPS-y-Koordinate des Kartenausschnitts	Dezimalzahl

Tabelle 6: Aufrufparameter der Schnittstelle Kartenausschnitt

Die Schnittstelle *GPSzuStraße* nimmt eine x- und eine y-Koordinate entgegen und liefert den Identifikationsschlüssel der Straße zurück, auf der dieser GPS-Punkt liegt (vgl. Tabelle 7). Falls der Punkt auf keiner Straße liegt, wird eine leere Zeichenkette zurückgegeben. Um Ungenauigkeiten der GPS-Geräte auszugleichen, kann hier eine Toleranz von einigen Metern berücksichtigt werden.

Bezeichnung	Beschreibung	Datentyp
x	GPS-x-Koordinate	Dezimalzahl
y	GPS-y-Koordinate	Dezimalzahl

Tabelle 7: Aufrufparameter der Schnittstelle GPSzuStraße

4.5.3 Schnittstelle zu Sensoren

Die Sensoren-Komponente stellt mittels einer ereignisbasierte Schnittstelle (*Sensorwert*, Tabelle 8) nach dem *Observer-Pattern* die Verbindung zu den Hardware-Sensoren her. Übergeordnete Komponenten können sich für einzelne Sensoren registrieren, deren Daten sie dann übermittelt bekommen. Die Hardware-spezifischen Sensordaten werden in ein für alle Sensoren einheitliches Format überführt (vgl. Abschnitt 4.4.2). Damit ist die Weitergabe auch bisher unbekannter Sensordaten innerhalb der KoSit-Komponente an neu entwickelte Verfahren der kooperativen Situationsanalyse möglich, ohne die KoSit-Komponente oder die Sensoren-Schnittstelle zu ändern. Die Daten werden als (Typ, Wert)-Tupel übergeben, der Typ identifiziert den Sensor und das Format bzw. die Einheit des Wertes.

28 In der aktuellen Version 0.6 von April 2009, siehe [48]

Bezeichnung	Beschreibung	Datentyp
key	Identifikationsschlüssel des Sensors	Zeichenkette
value	Aktueller Wert	Dezimalzahl

Tabelle 8: Rückgabeparameter der Schnittstelle Sensorwert

4.5.4 Schnittstelle zu GPS

Die GPS-Komponente ermöglicht die Abfrage der aktuellen Position als GPS-Punkt. Zurückgegeben werden die zugehörige x- und y-Koordinate.

4.5.5 Schnittstelle zu Kommunikation

Zum Austausch von Verkehrsinformationen und Sensordaten stellt die Kommunikationskomponente vier Schnittstellen bereit. Die Schnittstellen *AusgangSD* (Sensordatum) und *AusgangVI* (Verkehrsinformation) ermöglichen es anderen Komponenten, Sensordaten und generierte Verkehrsinformationen zu verschicken. Sensordaten enthalten neben dem Sensortyp und -wert auch die Position und den Zeitpunkt bei Erzeugung des Wertes. Ein Identifikationsschlüssel zur Erkennung mehrfach empfangener Daten ist hier nicht notwendig, da das Tupel (Zeitpunkt, Position) jeden Sensorwert eindeutig identifiziert. Verschickt werden können damit sowohl eigene als auch fremde, vorher empfangene Sensordaten.

Bezeichnung	Beschreibung	Datentyp
key	Identifikationsschlüssel des Sensors	Zeichenkette
value	Gemessener Sensorwert	Dezimalzahl
x	GPS-x-Koordinate	Dezimalzahl
y	GPS-y-Koordinate	Dezimalzahl
creation	Zeitstempel der Erzeugung des Sensorwertes	Zeitstempel

Tabelle 9: Parameter der Schnittstellen EingangSD/AusgangSD

Generierte Verkehrsinformationen können über die Schnittstelle *AusgangVI* verschickt werden. Diese orientiert sich an der Schnittstelle *Verkehrsinformation* der KoSit-Komponente, die um zwei Parameter erweitert wurde (vgl. Tabelle 10). Es muss der Identifikati-

onsschlüssel der Verkehrsinformation angegeben werden, um den mehrfachen Empfang derselben Verkehrsinformation erkennen zu können. Außerdem wird der Zeitpunkt der Erzeugung der Verkehrsinformation übertragen.

Bezeichnung	Beschreibung	Datentyp
id	Identifikationsschlüssel der Verkehrsinformation	Zeichenkette
type	Typ der Verkehrsinformation	Zeichenkette
roadId	Identifikationsschlüssel der betroffenen Straße	Zeichenkette
x1	x-Koordinate des Anfangs der Verkehrssituation	Dezimalzahl
y1	y-Koordinate des Anfangs der Verkehrssituation	Dezimalzahl
x2	x-Koordinate des Endes der Verkehrssituation	Dezimalzahl
y2	y-Koordinate des Endes der Verkehrssituation	Dezimalzahl
active	Status der Verkehrsinformation 1: gemeldete Verkehrssituation wurde erkannt 0: gemeldete Verkehrssituation besteht nicht mehr	binär
creation	Zeitstempel der Erzeugung der Verkehrsinformation	Zeitstempel

Tabelle 10: Parameter der Schnittstellen EingangVI/AusgangVI

Die Schnittstellen zum Empfangen von Verkehrsdaten sind analog zu den Schnittstellen für ausgehende Daten spezifiziert, *EingangSD* (Sensordatum) und *EingangVI* (Verkehrsinformation) nutzen die gleichen Parameter wie die Schnittstellen zum Ausgang der entsprechenden Verkehrsdaten (vgl. Tabellen 9 und 10). Eingegangene Daten leiten sie mittels *Observer-Pattern* an registrierte Komponenten weiter.

4.5.6 Zusammenfassung der Schnittstellen

In Abbildung 13 ist die KoSit-Komponente mit allen nach außen bereitgestellten und von anderen Komponenten benötigten Schnittstellen als Komponentendiagramm[29] dargestellt.

29 In UML2-Notation nach [60], S. 273 ff.

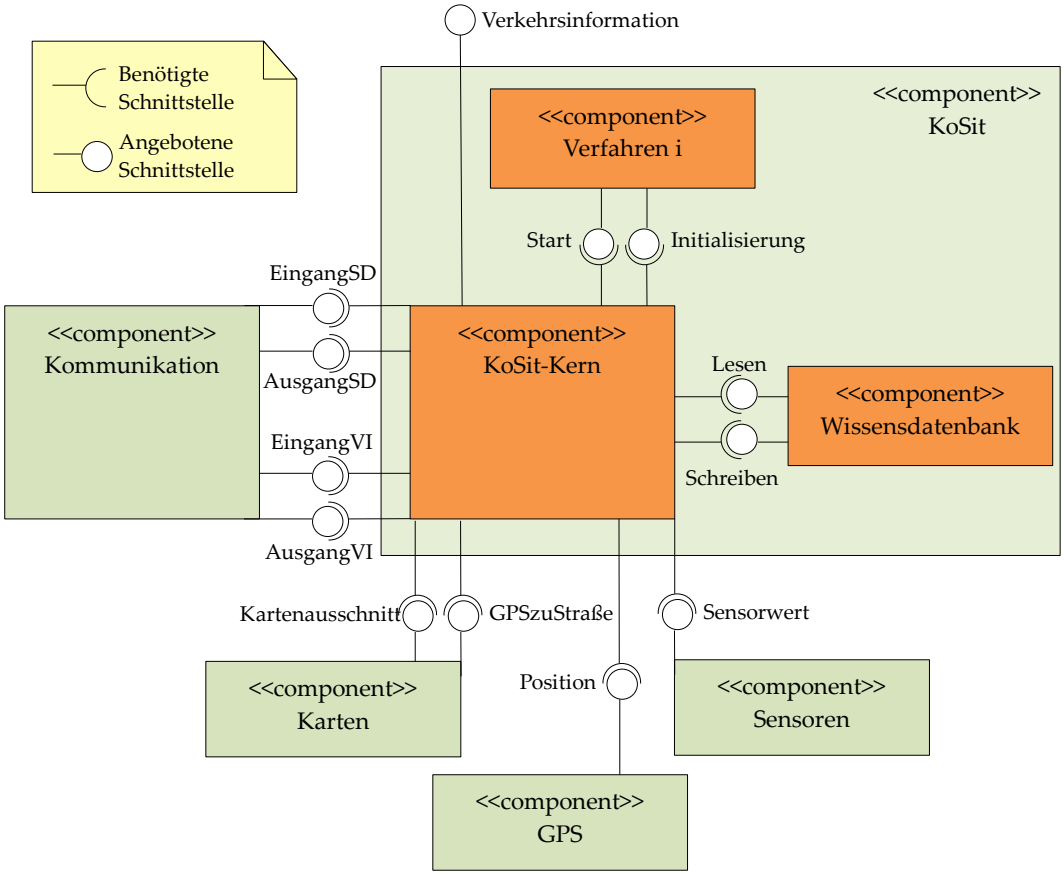

Abbildung 13: Komponentendiagramm KoSit

5 Spezifikation der Simulationsumgebung

5.1 Anforderungen an die Simulationsumgebung

Um die verschiedenen Verfahren der kooperativen Situationsanalyse evaluieren zu können, wird eine Simulationsumgebung konzipiert. In diesem Abschnitt werden die Anforderungen an die zu entwickelnde Simulationsumgebung beschrieben, aufgeteilt in funktionale und nicht-funktionale Anforderungen.

Funktionale Anforderungen

- **Mikroskopisches Verkehrsmodell:** Zur Auswertung der Sensordaten der Fahrzeuge ist die Modellierung einzelner Fahrzeuge notwendig. Demzufolge muss ein (sub-) mikroskopisches Modell verwendet werden.
- **Import und Simulation von Straßenverkehr:** Die Verfahren sollen mit möglichst realen Verkehrsdaten getestet werden. Dazu muss Straßenverkehr mit vorher vorgegebenen Merkmalen eingelesen und simuliert werden können.
- **Integration der KoSit-Komponente:** Die in Abschnitt 4.3 beschriebene KoSit-Komponente muss in die Simulation integriert werden können, um darin die Verfahren auf dem simulierten Straßenverkehr ausführen zu können. Dazu müssen die von der KoSit-Komponente benötigten Schnittstellen nachgebildet und an ihnen passende, simulierte Daten bereitgestellt werden.
- **Unterstützung verschiedener Verfahren:** Die Simulationsumgebung muss verschiedene Typen von Verfahren zur kooperativen Situationsanalyse unterstützen, die über die KoSit-Komponente ausgeführt werden.

- **Unterstützung von Stadt- und Autobahnverkehr:** OSSOTIS ist neben Autobahnen auch für innerstädtischen Verkehr konzipiert. Die Simulationsumgebung muss infolgedessen sowohl Stadt- als auch Autobahnverkehr unterstützen.
- **Parametrisierung der Anwendung:** Um die Simulation an verschiedene OSSOTIS-Implementierungen anzupassen, müssen die Reichweite der Funktechnik, der Verbreitungsgrad der OSSOTIS-Implementierung und die in den Fahrzeugen vorhandenen Sensoren über Parameter festgelegt werden können.
- **Anzeige der Ergebnisse:** Von einem Verfahren erkannte Verkehrsinformationen müssen dem Benutzer ausgegeben werden.

Nicht-funktionale Anforderungen

- **Hohe Simulationsgeschwindigkeit:** Für die Nutzung eines dezentralen Verkehrsleitsystems muss die Simulation viele Fahrzeuge enthalten, da nur ein Bruchteil der Fahrzeuge mit der Technik ausgestattet ist und das Straßennetz eine Ausdehnung über ein Vielfaches der Reichweite der verwendeten Funktechnologie haben sollte. Insbesondere zur Erzeugung und späteren Erkennung von Stausituationen ist eine hohe Anzahl an Fahrzeugen notwendig. Dies erfordert eine hohe Simulationsgeschwindigkeit, um in angemessener Zeit Ergebnisse zu erzielen.
- **Deterministische Simulation:** Um verschiedene Verfahren zu vergleichen, ist ein jeweils gleiches Fahrverhalten der simulierten Fahrzeuge notwendig. Das verwendete Modell des Straßenverkehrs sollte deshalb deterministisch sein.
- **Import von OSM-Kartenmaterial:** Die Verkehrssimulation sollte das auch von OSSOTIS genutzte OpenStreetMap-Kartenmaterial verwenden.
- **Nutzung von Open Source Technologien:** Das OSSOTIS-Projekt wird als Open Source System umgesetzt. Für die Nutzung und Verbreitung der dazugehörigen Simulationsumgebung ist es demnach von Vorteil, auch auf Open Source Lösungen zu basieren.

5.2 Architektur der Simulationsumgebung

5.2.1 Überblick

Die zu entwickelnde Simulationsumgebung lässt sich in zwei Funktionsbereiche aufteilen: die Simulation des Straßenverkehrs und die Simulation der kooperativen Situationsanalyse. Die Verkehrssimulation ist zuständig für die Bewegung der Fahrzeuge auf dem Straßennetz und die Bereitstellung der Sensordaten der simulierten Fahrzeuge. Die Simulation der kooperativen Situationsanalyse bildet die OSSOTIS-Architektur für die einzelnen Fahrzeug nach und führt die kooperative Situationsanalyse mit Hilfe der Verkehrsdaten aus der Simulation aus.

Für die Verkehrssimulation kann eines der in Abschnitt 2.5 vorgestellten Projekte genutzt werden. Je nach verwendetem Verfahren zur kooperativen Situationsanalyse müssen die Fahrzeuge in der Verkehrssimulation um die benötigten Sensoren erweitert werden. In dieser Arbeit wird als Beispielverfahren eine Stauerkennung implementiert, so dass keine weiteren Sensoren (neben denen für Position und Geschwindigkeit) vorhanden sein müssen. Im Folgenden wird SUMO verwendet, da es sich durch die freie Verfügbarkeit und die gut dokumentierte Schnittstelle besonders eignet. Außerdem wurde die Nutzbarkeit der Schnittstelle zur Erweiterung der Simulation um eigene Anwendungen in mehreren Forschungsprojekten belegt.[30] Das verwendete Fahrzeugfolge-Modell nach Krauß erfüllt die Anforderungen nach einem mikroskopischen Verkehrsmodell, der hohen Simulationsgeschwindigkeit und der deterministischen Simulation. SUMO erfüllt auch die anderen Anforderungen, wie in Abschnitt 5.2.3 gezeigt wird.

Für die Simulation der kooperativen Situationsanalyse wird die Anwendung *KSS* (*Kooperative Situationsanalyse – Simulation*) entwickelt. KSS simuliert die in Abschnitt 4.5 spezifizierten Schnittstellen, um die KoSit-Komponente und damit die eingebundenen Verfahren zur kooperativen Situationsanalyse auf den Daten der Verkehrssimulation ausführen zu können. Dies ist für jedes Fahrzeug in der Simulation, dass das OSSOTIS-System implementiert haben soll, notwendig. Dazu werden diese Fahrzeuge mit einer KoSit-Komponente in KSS modelliert und mit ihrer Repräsentation in der Verkehrssimulation verknüpft.

30 z.B. im eWorld-Projekt des Hasso Plattner Instituts, siehe [61]

Die Simulationsumgebung beinhaltet demzufolge die zu entwickelnde Anwendung KSS und die existierende Verkehrssimulation SUMO. Die Simulation in SUMO wird von KSS schrittweise gesteuert, um nach jedem Schritt die kooperative Situationsanalyse ausführen zu können (siehe Abbildung 14). KSS lässt sich durch Anpassung der Schnittstellenaufrufe auch mit anderen Verkehrssimulationen nutzen.

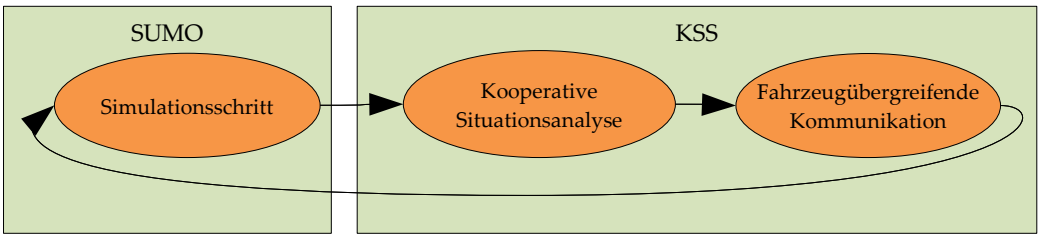

Abbildung 14: Aufteilung der Simulationsumgebung in KSS/SUMO

5.2.2 Ablauf der Simulation in KSS

Nach jedem Simulationsschritt der Verkehrssimulation wird die OSSOTIS-Funktionalität in KSS nachgebildet. Zuerst werden die Fahrzeugeigenschaften aus der Verkehrssimulation ausgelesen. Dazu gehören die Position, die Geschwindigkeit und mögliche weitere Sensorwerte. Diese Daten werden der KoSit-Komponente des zugehörigen Fahrzeugs über die Schnittstelle *Sensorwert* übergeben. Auf diesen und den bereits gespeicherten Daten werden die Verfahren zur kooperativen Situationsanalyse nach der Spezifikation in Abschnitt 4.3.1 (vgl. Abbildung 8) ausgeführt. Anschließend wird die Kommunikation zwischen den Fahrzeugen simuliert, indem parallel für alle Fahrzeuge die gespeicherten Sensordaten und Verkehrsinformationen per Broadcast an die empfangenden Fahrzeuge verteilt werden. Die Kommunikation ist durch die Simulationsparameter Funkreichweite und Funkbandbreite beschränkt. Der hier beschriebene Ablauf ist in Abbildung 15 als Aktivitätsdiagramm[31] dargestellt und wird für jedes in der Simulation vorhandene Fahrzeug ausgeführt.

31 In UML2-Notation nach [51]

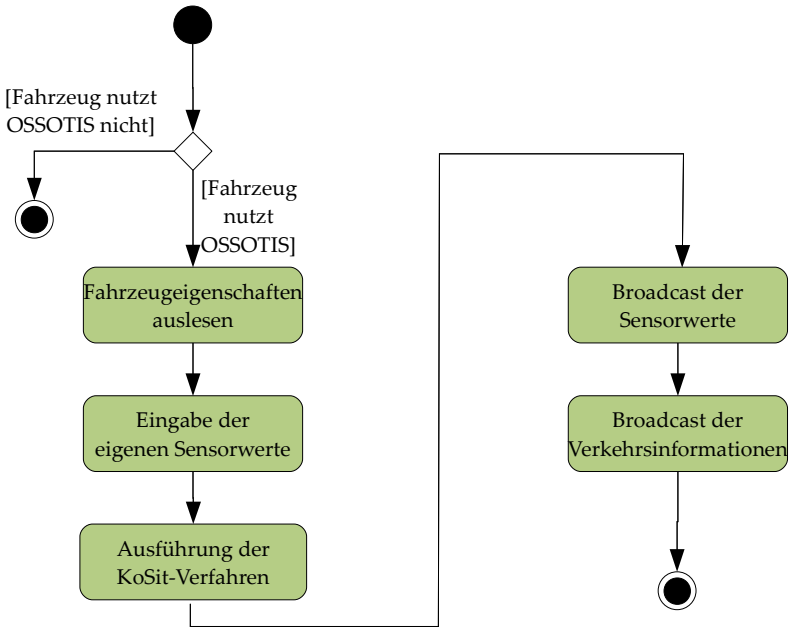

Abbildung 15: Aktivitätsdiagramm zum Ablauf von KSS

5.2.3 Nutzung von SUMO

Dieser Abschnitt beschreibt die für die Simulationsumgebung notwendigen Funktionen von SUMO: Die Simulation von Verkehrsszenarien bestehend aus Straßennetzen und sich darauf bewegenden Fahrzeugen, die Erweiterung der von SUMO simulierten Fahrzeuge um Fahrzeugsensoren und die Schnittstelle zur Steuerung von SUMO durch KSS.

Simulation von Verkehrsszenarien

Für die Evaluation der Verfahren zur kooperativen Situationsanalyse ist es wichtig, vorgegebene Verkehrsszenarien auf realistischen Straßennetzen simulieren zu können. Die SUMO-Teilanwendung NETCONVERT ermöglicht die Konvertierung von Straßennetzen in das SUMO-spezifische XML-Format, dass das Straßennetz als Graph enthält. Importiert werden können neben den Dateiformaten verschiedener Simulationsumgebungen auch OpenStreetMap-Straßenkarten. Der gewünschte Ausschnitt kann dazu über die Webservice-Schnittstelle der OSM-API abgerufen werden.

Auf dem so generierten Straßennetz können Fahrzeugen an Hand von so genannten *Trips* bewegt werden. Diese legen die Start- und Zielposition fest, zwischen denen sich ein Fahrzeug auf dem Straßennetz bewegen soll. Dazu ist die Festlegung einer *Route* notwendig, in der der Weg durch alle zwischen Start und Ziel liegenden Knoten spezifiziert ist. Diese Route kann auf mehrere Arten festgelegt werden: durch den Import bestehender Routen aus anderen Anwendungen, die Generierung durch eine der SUMO-Anwendungen oder dynamisch während der Simulation. Die dynamische Routenwahl berücksichtigt das Verkehrsaufkommen auf den für das Fahrzeug gewählten Straßen und ändert gegebenenfalls die Route, wenn das Fahrzeug dadurch schneller zum Ziel gelangt. Für die Simulationsumgebung sind je nach verwendetem Verfahren zur kooperativen Situationsanalyse sowohl die dynamische Routenwahl als auch die vorherige Generierung fester Routen geeignet. Zur Stauerkennung kann die dynamische Routenwahl nicht eingesetzt werden, da die Fahrzeuge hierbei Stausituationen nach Möglichkeit vermeiden und diese somit gar nicht erst entstehen. Infolgedessen können sie auch nicht erkannt werden.

Erweiterung um Fahrzeugsensoren

Die in SUMO simulierten Fahrzeuge besitzen die Eigenschaften Position und Geschwindigkeit, die von externen Anwendungen abgerufen werden können. Weitere Sensoren sind nicht vorgesehen. Zur Erkennung von weiteren Verkehrssituationen neben Staus müssten die dafür nötigen Sensoren in die Simulation integriert werden. Das ist durch den frei verfügbaren Quellcode zwar möglich, aber sehr aufwändig. Eine derartige Erweiterung ist allerdings nicht Teil dieser Arbeit.

Schnittstelle TraCI

Die TraCI-Schnittstelle erlaubt den Eingriff in die SUMO-Simulation durch externe Anwendungen. Beim Start wird SUMO mitgeteilt, als Server zu agieren und auf einem bestimmten TCP-Port auf Befehle zu warten anstatt mit der Simulation zu beginnen. Der Client kann über Datenpakete die Simulation starten, anhalten,(schrittweise) fortführen, Einfluss auf das Verhalten einzelner Fahrzeuge nehmen oder Eigenschaften der Simulation oder von Fahrzeugen abfragen. Die einzelnen Befehle werden über einen *Identifier* unterschieden, der mit einem zusätzlichen Parameter den Inhalt des Datenpakets bildet. Die Antwort von SUMO besteht aus einem *Statuscode* (erfolgreich/fehlgeschlagen/nicht implementiert) und den vom Befehl abhängigen zusätzlichen Daten.

Für die Verbindung von KSS und SUMO wird die „zweite Generation“ der Schnittstelle verwendet, bei der die Befehle durch *String-Identifier* gekennzeichnet sind. Eine vollständige Beschreibung der TraCI-Schnittstelle findet sich in der SUMO-Dokumentation in [45].

5.2.4 Spezifikation KSS

In diesem Abschnitt wird die OSSOTIS-Simulation KSS spezifiziert. Dazu werden alle erforderlichen Funktionen und das Zusammenspiel mit SUMO beschrieben. KSS besitzt eine Benutzerschnittstelle, über die die Parameter der Simulation übergeben werden können und die Ergebnisse der kooperativen Situationsanalyse angezeigt werden.

Simulationsparameter

Neben dem von SUMO simulierten Straßennetz und Verkehrsaufkommen muss die Simulation weiter parametrisiert werden können. Die folgenden Parameter werden dazu bereitgestellt:

- **Funkreichweite r:** Über die Funkreichweite lässt sich einstellen, welche Fahrzeuge miteinander kommunizieren können. Die Funkreichweite entspricht dem maximalen Abstand zweier kommunizierender Fahrzeuge unabhängig von der Straßenführung. Dieser Parameter muss in Abhängigkeit von der zu simulierenden Funktechnik bestimmt werden. Geographische Besonderheiten oder Abschattungen durch Gebäude werden nicht berücksichtigt, die reale Funkreichweite kann also von dem hier verwendeten Kreis mit dem Radius r abweichen.
- **Funkbandbreite b:** Die Funkbandbreite begrenzt den Umfang der Fahrzeugkommunikation. Die Fahrzeuge übertragen die Verkehrsdaten per Broadcast an alle Fahrzeuge in der Funkreichweite und beeinträchtigen damit die Übertragung anderer Fahrzeuge. Daher ist die Funkbandbreite für jedes Fahrzeug in der Realität nicht einheitlich, sondern hängt von der Fahrzeugdichte und dem Medienzugriff der anderen Fahrzeuge ab. In der Simulation wird dieser Sachverhalt vereinfacht, indem die Funkbandbreite für jedes Fahrzeug auf einen festen Wert b eingestellt wird, der den maximal möglichen Umfang an zu übertragender Verkehrsdaten bestimmt.

- **OSSOTIS-Verbreitungsgrad v:** Mit diesem Parameter kann angegeben werden, welcher Anteil der Fahrzeuge mit der OSSOTIS-Technik ausgestattet sein soll. KSS simuliert nur für diesen Teil der Fahrzeuge aus SUMO die OSSOTIS-Funktionen. Der Wertebereich für v liegt zwischen 0 (kein Fahrzeug nutzt OSSOTIS) und 1 (alle Fahrzeuge nutzen OSSOTIS). Die Fahrzeuge werden zufällig ausgewählt.

Steuerung von SUMO

KSS steuert die Verkehrssimulation in SUMO über die TraCI-Schnittstelle (siehe Abschnitt 5.2.3). Mit dem Befehl *Simulation Step* wird der nächste Schritt in der Simulation ausgeführt. Anschließend wird eine Liste aller in SUMO vorhandenen Fahrzeuge abgerufen, dies erfolgt durch den Befehl *Get Vehicle Variable* mit dem Parameter *id list*. Für die Fahrzeug-IDs, für die KSS die OSSOTIS-Funktionen simuliert hat, werden die Fahrzeugdaten mit dem Befehl *Get Vehicle Variable* und dem zu dem jeweiligen Sensor passenden Parameter (*speed*, *position*, ggf. weitere) abgerufen.

Modellierung der Fahrzeuge

KSS muss intern eine Instanz der KoSit-Komponente für jedes Fahrzeug in der Verkehrssimulation von SUMO ausführen, dass die OSSOTIS-Funktionalität integriert haben soll. Die KoSit-Komponente muss mit dem jeweiligen Fahrzeug in SUMO verknüpft sein, damit sich die durch die Simulation generierten Sensordaten zuordnen lassen. Die Zuordnung erfolgt über den von SUMO vergebenen Identifikationsschlüssel, der beim Abruf der Fahrzeugliste übergeben wird und auch zum Zugriff auf die Fahrzeugeigenschaften benötigt wird.

Simulation der Kommunikation

Die Kommunikation läuft für alle Fahrzeuge parallel ab. Für jedes Fahrzeug f werden alle Fahrzeuge mit simulierter OSSOTIS-Technik und einem Abstand kleiner der Funkreichweite ermittelt. KSS übergibt jedem dieser Fahrzeug die vom Fahrzeug f über die Kommunikationsschnittstellen übergebenen Sensordaten und Verkehrsinformationen an die Schnittstellen *EingangSD* bzw. *EingangVI* der KoSit-Komponente. Die Menge der jeweils „übertragenen“ Verkehrsdaten wird dabei durch die Funkbandbreite begrenzt. Die Kommunikation wird nur einmal pro Simulationsschritt simuliert, die von den KoSit-Kompo-

nenten periodisch an die Kommunikationsschnittstelle übergebenen Verkehrsdaten werden von KSS gesammelt. Dies ist erforderlich, da die Simulation zentral gesteuert wird und SUMO rundenbasiert fortgesetzt werden muss.

Bereitstellung des Kartenmaterials

Das Kartenmaterial wird den Verfahren zur kooperativen Situationsanalyse mit Hilfe des OpenStreetMap-Webservice bereitgestellt. KSS ruft dazu den benötigten Kartenausschnitt über die API ab und übergibt ihn an das Verfahren. Möglich ist hier auch die lokale Speicherung des gesamten für die Simulation benötigten Kartenmaterials, um Geschwindigkeitseinbußen durch den Zugriff auf den Webservice zu vermeiden. Der benötigte Kartenausschnitt ist durch die Verwendung als Straßennetz in SUMO festgelegt und damit im Vorfeld bekannt.

Ausgabe der Ergebnisse

Die Ergebnisse der kooperativen Situationsanalyse in Form von Verkehrsinformationen werden von KSS dem Benutzer angezeigt. KSS registriert sich bei den Schnittstellen *Verkehrsinformation* aller simulierten KoSit-Komponenten und erhält daraufhin die generierten Verkehrsinformationen. Die in Verkehrsinformationen enthaltenen Datenfelder sind in Abschnitt 4.4.1 beschrieben. Ausgegeben werden der Typ der Verkehrsinformation, die betroffene Straße, die Anfangs- und Endposition und das Fahrzeug, das die Verkehrsinformation generiert hat.

Architektur KSS

Die oben beschriebene Funktionalität von KSS wurde auf drei Komponenten verteilt: die Nachbildung der OSSOTIS-Schnittstellen zur Integration der KoSit-Komponente, die Kapselung der Schnittstelle zur Verkehrssimulation und die Simulationssteuerung. Die OSSOTIS-Schnittstellen der Karten- und GPS-Komponente werden den Verfahren über eine Klasse zur Verfügung gestellt, die die erforderlichen Daten über den OpenStreetMap-Webservice beziehungsweise aus der Verkehrssimulation bezieht. Die Kommunikationsschnittstellen sprechen direkt die Simulationssteuerung an, die die Kommunikation zwischen den einzelnen von ihr modellierten KoSit-Komponenten intern simuliert. Außerdem steuert sie die Verkehrssimulation.

Die Kapselung der Anbindung der Verkehrssimulation erlaubt den einfachen Austausch von SUMO gegen andere Verkehrssimulationen mit einer geeigneten Schnittstelle oder auch den Vergleich verschiedener Verkehrssimulationen. Die TCP-Kommunikation erfolgt innerhalb dieser bereitgestellten Methoden, die die jeweils passenden Befehle an SUMO schicken und die Antwort zurückgeben:

- `step()`[32]
  Mit dieser Methode wird die Verkehrssimulation um einen Schritt fortgesetzt. Diese Methode wird von der Simulationsumgebung zum Start und nach jedem Simulationslauf aufgerufen.
- `getIds(): String[]`
  Ruft die Identifikationsschlüssel aller in der Verkehrssimulation vorhandenen Fahrzeuge ab.
- `getPosition(String id): int[]`
  Ruft die Position eines durch den Parameter identifizierten Fahrzeuges ab. Zurückgegeben werden die x- und y-GPS-Koordinaten.
- `get(String id, String variable): String`
  Ruft den Wert einer beliebige Fahrzeugvariable für ein durch den Parameter *id* identifiziertes Fahrzeug ab.
- `stop()`
  Beendet die Verkehrssimulation.

5.2.5 Spezifikation des Beispiel-Verfahrens

Das hier vorgestellte Verfahren zur Erkennung von Stausituationen dient nur der Demonstration der Funktionsweise der Infrastruktur zur kooperativen Situationsanalyse und ist hinsichtlich der Erkennungsrate und des Laufzeitverhaltens verbesserungswürdig. Die Entwicklung effizienterer Verfahren kann im Rahmen dieser Arbeit aber nicht geleistet werden.

Der Algorithmus nutzt die Fahrzeugeigenschaften *Position* und *Geschwindigkeit*. Die Erkennung eines Staus erfolgt durch die Aggregation der Geschwindigkeiten in einem bestimmten Bereich zu einer mittleren Geschwindigkeit. Liegt diese unter 5 km/h und befinden sich mindestens zehn (mit OSSOTIS ausgestattete) Fahrzeuge in diesem Bereich, generiert das Verfahren eine Verkehrsinformation des Typs *Stau* (siehe Abbildung 16). Die Begrenzung auf mindestens zehn Fahrzeuge verhindert die Falschmeldung eines Staus, wenn lediglich ein Fahrzeug aus anderen Gründen mit niedriger Geschwindigkeit

32 Pseudocode, Syntax: `Methodenname([Parameter])[: Rückgabe]`

in diesem Bereich unterwegs sind. Zur Abgrenzung der Bereiche wird das vorliegende Kartenmaterial in Quadrate mit 100m Kantenlänge unterteilt und für jede darin befindliche Straße die mittlere Geschwindigkeit berechnet. Ist eine Staumeldung für einen Bereich vorhanden, die aber an Hand der vorliegenden Sensordaten nicht verifiziert werden kann (also keine bestätigende Verkehrsinformation generiert wurde), wird diese Staumeldung durch die Generierung einer negativen Verkehrsinformation widerrufen.

Eine mögliche Verbesserung des Verfahrens um auch stockenden Verkehr zu erkennen könnte typische Geschwindigkeiten für die verschiedenen Straßentypen berücksichtigen und bei Unterschreitung dieser durch die ermittelte mittlere Geschwindigkeit um beispielsweise mehr als 30% eine entsprechende Verkehrsinformation generieren.

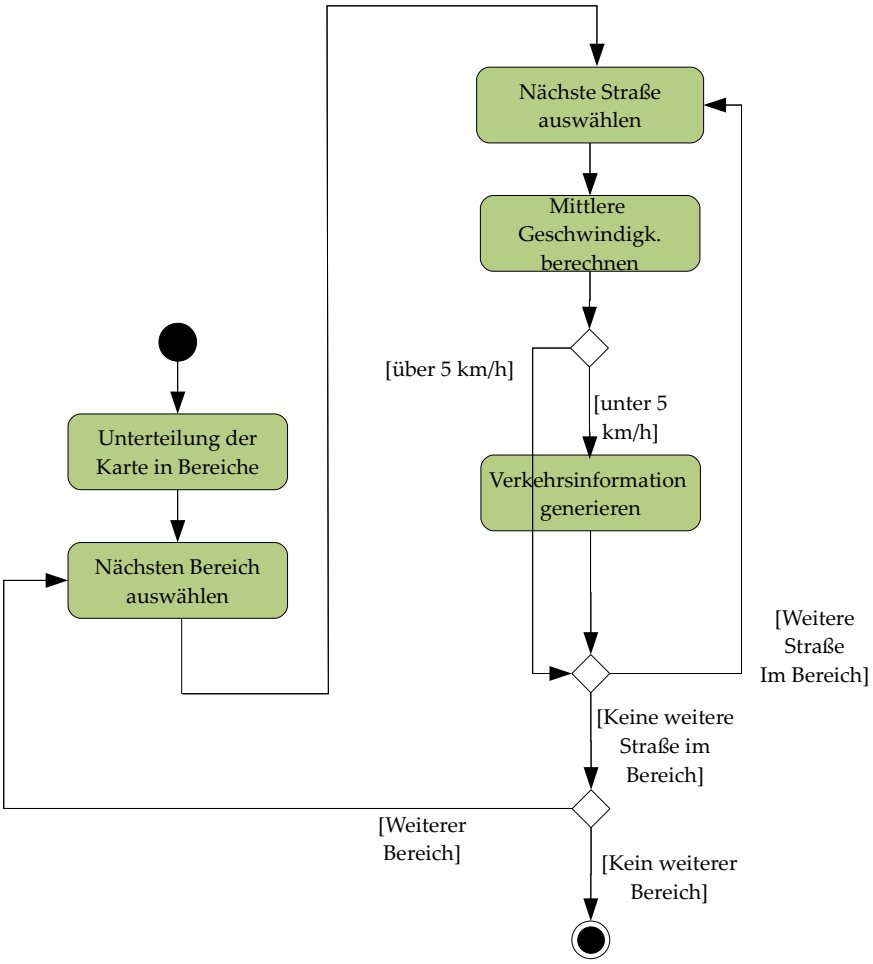

Abbildung 16: Aktivitätsdiagramm Stauerkennung

6 Implementierung

6.1 Überblick

In diesem Kapitel wird die prototypische Implementierung der in der vorliegenden Arbeit entwickelten Komponenten genauer betrachtet. Nach der Vorstellung der verwendeten Entwicklungswerkzeuge werden die einzelnen implementierten Teilbereiche beschrieben, angefangen mit der KoSit-Komponente bestehend aus KoSit-Kern, Wissensdatenbank und der Schnittstelle zu den Verfahren. Anschließend wird auf die Komponente KSS der Simulationsoberfläche eingegangen und die Umsetzung des Verfahrens zur Stauerkennung kurz erläutert.

Weitergehende Informationen zu den einzelnen Komponenten und den hier nicht beschriebenen internen Klassen finden sich in der *javadoc*-Dokumentation. In Anhang B sind die vollständigen Klassendiagramme von KSS und der KoSit-Komponente abgebildet. In Anhang A finden sich Hinweise zur Einrichtung und Nutzung von SUMO und der Simulationsumgebung KSS.

6.2 Auswahl der Entwicklungswerkzeuge

Das OSSOTIS-Projekt nutzt für die Integration der einzelnen Komponenten zu einem Gesamtsystem die auf Java basierende OSGi-Ausführungsplattform. Da die Software auch auf eingebetteten Systemen in Fahrzeugen installiert werden soll, ist eine möglichst große Portabilität notwendig, welche mit Java gegeben ist. Die KoSit-Komponente soll innerhalb der OSGi-Plattform ausgeführt werden und wird deshalb ebenfalls in Java implementiert. Sämtliche Komponenten werden objektorientiert in Form von Klassen modelliert, um die spätere Erweiterung der Software zu vereinfachen.

Die Verkehrssimulation SUMO wurde in C++ geschrieben. Da der Zugriff auf die Simulationseigenschaften über TCP erfolgt und SUMO als eigenständige Anwendung ausgeführt wird, stellt dies kein Problem dar.

6.3 Beschreibung der entwickelten Komponenten

6.3.1 Implementierungsdetails der Wissensdatenbank

Die Wissensdatenbank nutzt eine eingebettete SQLite-Datenbank zur Speicherung der Daten. Dies ermöglicht den einfachen Einsatz in eingebetteten Systemen, auf denen herkömmliche, eigenständige relationale Datenbanksysteme nicht lauffähig sind. Das Datenmodell enthält die zwei Entitäten *Sensordatum* und *Verkehrsinformation*, die durch Spezialisierung von der Entität *Verkehrsdatum* abgeleitet werden. Die Attribute entsprechen den in den Abschnitten 4.4 festgelegten Datenfeldern der jeweiligen Datenstruktur. Das Datenmodell ist in Abbildung 17 als *Entity-Relationship-Modell* dargestellt.

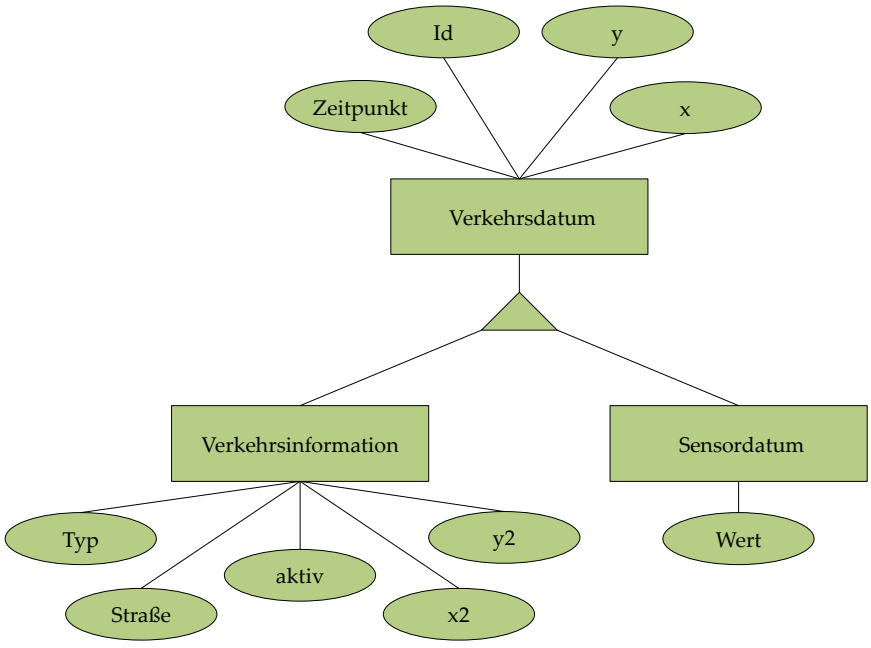

Abbildung 17: Entity-Relationship-Modell zur Wissensdatenbank

Der Filter zur Selektion der aus der Wissensdatenbank abgerufenen Verkehrsdaten wurde als eigenständige Klasse implementiert, die Eigenschaften für die Eingrenzungsmöglichkeiten (Alter des Verkehrsdatums, Erzeugung in fremdem/lokalem Fahrzeug, Typ Sensordatum, Typ Verkehrsinformation) besitzt.

6.3.2 Implementierungsdetails der Verfahrens-Schnittstelle

Die Schnittstelle zu den einzelnen Verfahren der kooperativen Situationsanalyse wurde über eine abstrakte Klasse *Verfahren* realisiert, von der die Verfahren abgeleitet werden müssen. Darin enthalten sind zwei Attribute für die Verkehrsdatenliste und die Datenbank sowie eine Methode zum Abruf der Verkehrsdatenliste. Die beiden abstrakten Methoden *init* und *start* müssen vom Verfahren durch Überlagerung implementiert werden. *start* beinhaltet die Programmlogik der kooperativen Situationsanalyse.

6.3.3 Implementierungsdetails der Simulationsoberfläche

Das zur Simulation des Verkehrs genutzte Programmpaket SUMO wurde nicht verändert. Die OSSOTIS-Simulation KSS wurde entsprechend der Spezifikation in Abschnitt 5.2.4 implementiert. Da die OSGi-Plattform des OSSOTIS-Gesamtsystems nicht verfügbar ist, wurden die Schnittstellen der KoSit-Komponente direkt über Methodenaufrufe der Objektinstanzen angesprochen. Die benötigten OSSOTIS-Objekte werden als Parameter im Konstruktor übergeben.

Das in Abschnitt 5.2.5 spezifizierte Verfahren zur Stauerkennung wurde exemplarisch implementiert um die Nutzbarkeit des Systems zu beweisen. Die benutzte Verkehrsdatenliste enthält dabei diese Werte: *Stau* als Typ der generierten Verkehrsinformation, *Position* und *Geschwindigkeit* als relevante Verkehrsdaten mit jeweils 30 Sekunden Wartezeit.

6.4 Implementierung von Testszenarien

Zur Evaluation verschiedener Verfahren der kooperativen Situationsanalyse werden auf das jeweilige Verfahren angepasste Testszenarien bestehend aus einer Straßenkarte und darauf zu simulierendem Fahrzeugverkehr benötigt. Insbesondere die Größe des Straßennetzes und die Verkehrsdichte sind verfahrensspezifisch zu wählen. Der technische Ablauf zur Erstellung und Ausführung von Testszenarien ist in Anhang A beschrieben.

Zur Evaluation der Simulationsumgebung und des Beispielverfahrens wurden mehrere Verkehrsszenarien entwickelt. Als Kartenbasis wurden zwei verschiedene Ausschnitte des Straßennetzes von Dortmund ausgewählt, ein ca. 20 km² großer Bereich (siehe Abbildung 18) sowie ein auf das Umfeld des Universitätsgeländes beschränkter Bereich. Abbildung 19 stellt das größere Netz nach der Konvertierung als Screenshot in SUMO dar. Auf beiden Straßennetzen wurden jeweils zwei Verkehrssituationen mit 50 und 200 bzw. 100 und 1000 Fahrzeugen erstellt. Die Routen durch das Straßennetz wurden mit Hilfe der SUMO-Anwendung *duarouter* zufällig für die einzelnen Fahrzeuge generiert.

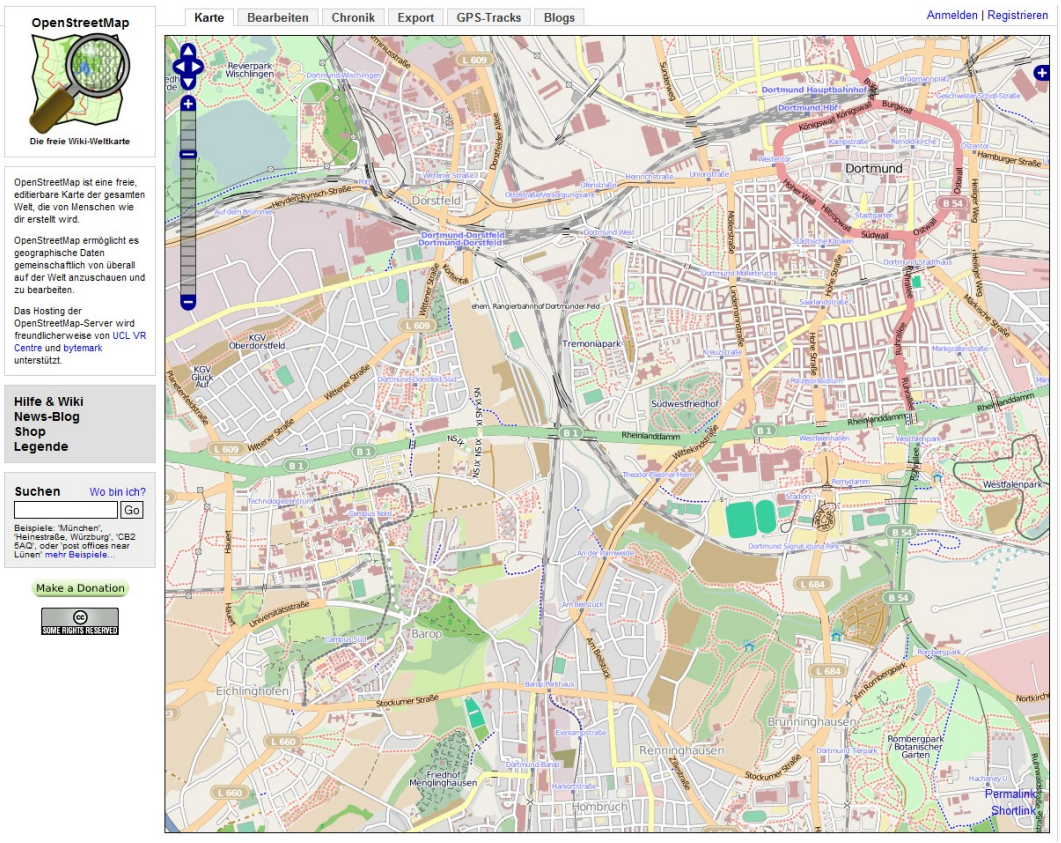

Abbildung 18: Screenshot OpenStreetMap

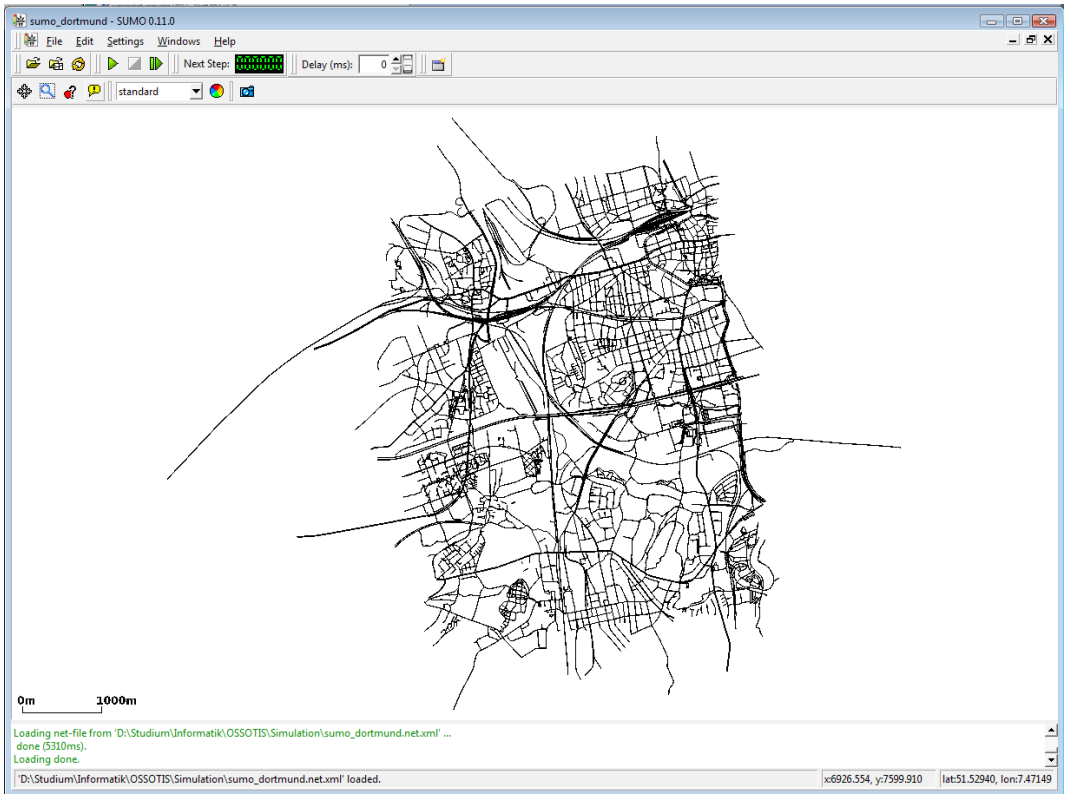

Abbildung 19: Screenshot SUMO

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

Das Ziel dieser Diplomarbeit war die Entwicklung einer Infrastruktur, mit der die kooperative Situationsanalyse in Form verschiedener Verfahren in ein dezentrales Verkehrsleitsystem (SOTIS) integriert werden kann. Den konkreten Rahmen hierfür bildete das OS-SOTIS-Projekt, das eine offene SOTIS-Architektur bereitstellen soll. Aufbauend auf diese Infrastruktur sollte eine Simulationsumgebung entwickelt werden, in der die Verfahren unabhängig vom gesamten SOTIS-System evaluiert werden können.

Neben technischen Grundlagen wurde der aktuelle Stand der Forschung im Bereich der dezentralen Verkehrsleitsysteme erarbeitet und der jeweilige Ansatz zur kooperativen Situationsanalyse beschrieben. Außerdem wurden geeignete Verkehrssimulationen vorgestellt. Anschließend wurden die Ziele und Grundzüge des OSSOTIS-Projekts sowie die Rolle der kooperativen Situationsanalyse darin betrachtet.

Durch die von OSSOTIS gegebenen Rahmenbedingungen und die Ziele dieser Arbeit ergaben sich eine Reihe von Anforderungen an das zu entwickelnde System. Diese wurden identifiziert und davon ausgehend die Infrastruktur für die kooperative Situationsanalyse in Form der KoSit-Komponente konzipiert. Hierbei wurde besonders auf den fahrzeugübergreifenden (*kooperativen*) Aspekt der Analyse der Verkehrssituationen und die flexible Unterstützung verschiedener Verfahren eingegangen. Außerdem wurden die benötigten Daten beschrieben und die Schnittstellen zu den anderen Komponenten des Gesamtsystems spezifiziert. Nach der Beschreibung der Anforderungen an die Simulationsumgebung wurde diese aufbauend auf die KoSit-Komponente und eine existierende Lösung zur Verkehrssimulation (SUMO) entwickelt. Der Fokus lag an dieser Stelle auf der von der OSSOTIS-Architektur losgelösten Ausführung der kooperativen Simulationsumgebung und der Einbindung der Verkehrssimulation. Abschließend wurde kurz auf die Besonderheiten bei der prototypischen Implementierung der erarbeiteten Konzepte eingegangen.

In bisherigen Forschungsprojekten zu dezentralen Verkehrsleitsystemen wurden hauptsächlich die technischen Randbedingungen und die Gesamtarchitektur der Systeme betrachtet. Die Analyse von Verkehrssituationen erfolgte meist in einer „Blackbox“ (vgl. Abschnitt 2.4). In dieser Arbeit wurde die kooperative Situationsanalyse als Kernkomponente der dezentralen Verkehrsleitsysteme detailliert betrachtet und damit ein bisher vernachlässigter Bereich ausführlich bearbeitet. Mit der entwickelten Simulationsumgebung können Verfahren zur kooperativen Situationsanalyse zudem einfach und einheitlich evaluiert werden. Dies ermöglicht den Vergleich unterschiedlicher Verfahren, deren Optimierung und damit die Entwicklung effizienter Verfahren.

Als Teilstück des OSSOTIS-Projekt betrachtet bringt das vorliegende Konzept die einheitliche Standardisierung eines offenen SOTIS als Gegenstück zu den bisherigen, in der Regel von der Automobilbranche unterstützten, proprietären Lösungen voran. Hierzu ist allerdings noch die Integration der bisherigen Teilkonzepte zu einem Gesamtsystem und die Entwicklung der bisher nicht betrachteten Teile notwendig.

7.2 Ausblick

Dezentrale Verkehrsleitsysteme und damit auch die kooperative Erkennung von Verkehrssituationen sind ein weites Forschungsfeld, das in einer Diplomarbeit nicht abschließend bearbeitet werden kann. Das hier entwickelte Konzept stellt die Infrastruktur für Verfahren der kooperativen Situationsanalyse dar. Auf dieser Basis müssen nun effiziente Verfahren zur Erkennung verschiedener Verkehrssituationen entwickelt werden, um die Vorteile dieser flexiblen Infrastruktur und allgemein der dezentralen Verkehrsleitsysteme gegenüber herkömmlichen, Infrastruktur-basierten Verkehrsleitsystemen zu beweisen.

Bei der Simulation von Verfahren kann es je nach zu erkennender Verkehrssituation zu Problemen mit den benötigten Sensorwerten kommen. Die Verkehrssimulation SUMO lässt sich zwar prinzipiell erweitern, das verwendete Verkehrsmodell stellt aber nicht alle in realen Fahrzeugen vorhandenen Daten bereit. Hier könnte die Nutzung eines submikroskopischen Verkehrsmodells in Betracht gezogen werden. Bei der Simulation der Fahrzeugkommunikation wurden typischen Störeffekte der Funkübertragung nicht betrachtet, für realistischere Ergebnisse sollten diese mit einbezogen werden. Neben den Ergebnissen aus Simulationen ist anschließend auch die reale Erprobung der Systeme wichtig, um in Simulationen nicht betrachtete oder nicht modellierbare Probleme erkennen und beseitigen zu können. Hierfür müssen in der Zukunft breit angelegte Feldversuche (wie im kürzlich gestarteten Projekt simTD) die Nutzbarkeit der kooperativen Situationsanalyse in der Praxis beweisen.

Literaturverzeichnis

1. *unbekannt*: Traffic and Traveller Information (TTI) - TTI messages via Traffic message coding - Part 2: Event and information codes for Radio Data System - Traffic Message Channel (RDS-TMC). ISO 14819-2

2. *Lars Wischof, André Ebner, Hermann Rohling, Matthias Lott, Rüdiger Halfmann*: SOTIS - A Self-Organizing Traffic Information System. Technical University of Hamburg-Harburg und Information and Communication Mobile (Siemens AG), München 2004

3. Projektdokumentation Network On Wheels. http://www.network-on-wheels.de, geprüft: 31.08.2009

4. *Martin Mauve, Walter Franz, Hannes Hartenstein*: Inter-Vehicle-Communications Based on Ad Hoc Networking Principles - The FleetNet Project. Universitätsverlag, Karlsruhe 2005

5. *Karl-Heinz Dietsche, Robert Bosch GmbH*: Kraftfahrtechnisches Taschenbuch. Vieweg, Wiesbaden 2007

6. *Robert Bosch GmbH*: Sensoren im Kraftfahrzeug. Christiani, Konstanz 2001

7. *Christian Adler*: Information Dissemination in Vehicular Ad Hoc Networks. Institut für Informatik, Ludwig-Maximilians-Universität, München 2006

8. *Jochen Schiller*: Mobilkommunikation. Pearson Studium, München 2003

9. *Timo Kosch*: Technial Concept And Prerequisites of Car-To-Car Communication. 5th European Congress and Exhibition on ITS 2006

10. *Wilfried Enkelmann*: FleetNet - Applications for Inter-Vehicle Communication. Intelligent Vehicles Symposium, Proceedings. IEEE 2003

11. *Christian Adler*: Information Disseminationin Vehicular Ad Hoc Networks. Institut für Informatik, Ludwig-Maximilians-Universität, München 2006

12. *Markus Straßberger, Christian Adler*: Lokale Gefahrenwarnung in Fahrzeug-Ad-Hoc-Netzen – Eine umfassende Analyse und aktuelle Lösungsansätze. BMW Group Forschung und Technik, München 2006

13. *Walter Franz*: Car-to-Car Communication - Anwendungen und aktuelle Forschungsprogramme in Europa, USA und Japan. Fachtagungsbericht GMM, VDE-Kongress, Berlin 2004

14. *Manuel Reil*: Entwurf einer Sicherheitsinfrastruktur für Vehicular Ad-hoc Networks (VANETs). Universität Regensburg, Wirtschaftswissenschaftliche Fakultät, Regensburg 2006

15. *Hermann Bischl, E. Lutz, W. Schaeffer*: Abschlussbericht zum EUREKA-Projekt PROMETHEUS. Abschlussbericht zum EUREKA-Projekt PROMETHEUS fuer die Deutsche Aerospace A1 E1/HA 1992

16. *Florian Dötzer, Markus Strassberger, Timo Kotsch*: Classification for traffic related inter-vehicle messaging. BMW Group Research and Technology, München 2005

17. *Maximilian Reiß*: Fusion of Spatio-Temporal Information and Knowledge in Vehicles using Probabilistic Networks. Universität Passau, Fakultät für Mathematik und Informatik, Passau 2006

18. *DEUFRAKO Project Consortium*: IVHW System Concept and issues relevant for standardization. DaimlerCrysler AG, Stuttgart 2002

19. Kurzdarstellung des Teilprojektes Verkehrsleistungsassistenz VLA als ergänzende Information zur INVENT-Broschüre. http://invent-online.de/downloads/VLA-handout-D.pdf, geprüft: 25.08.2009

20. *Hariharan Krishnan*: Vehicle Safety Communications Project. Vehicle Safety Communications Consortium 2006

21. *Lars Wischof, André Ebner, Hermann Rohling*: Self-Organizing Traffic Information System based on Car-to-Car Communication: Prototype Implementation. 1st International Workshop on Intelligent Transportation, Hamburg 2004

22. *Lars Wischof, André Ebner, Hermann Rohling, Matthias Lott, Rüdiger Halfmann*: SOTIS - A Self-Organizing Traffic Informatino System. Technical University of Hamburg-Harburg und Information and Communication Mobile (Siemens AG), München 2004

23. *A. Festag, G. Noecker, M. Strassberger, A. Lübke, B. Bochow, M. Torrent-Moreno, S. Schnaufer, R. Eigner, C. Catrinescu, J. Kunisch*: NoW - Network on Wheels: Project Objectives, Technology and Achievements. In proceedings of 6th International Workshop on Intelligent Transportation (WIT), Hamburg 2008

24. *Markus Straßberger*: Cooperative Active Safety Applications. NoW - Network on Wheels: Final Workshop, Ulm 2008

25. *Matthias Schulze, Gerhard Noecker, Konrad Boehm* : PReVENT: A European program to improve active safety. 5th International Conference on Intelligent Transportation Systems Telecommunications, Brest, Frankreich 2005

26. *Roberto Brignolo, Luisa Andreone, Michele Provera*: SAFESPOT Integrated Project. Co-operative Systems for Road Safety, "Smart Vehicles on Smart Roads". Centro Ricerche Fiat, Italien 2006

27. *unbekannt*: CAR 2 CAR Communication Consortium Manifesto: Overview of the C2C-CC System. CAR 2 CAR Communication Consortium 2007

28. *Christian Weiß, Helen Däuwel*: simTD Newsletter 001. simTD Konsortium 2009

29. *Panos Pavlidis*: Werkzeuge für die Verkehrssimulation. Institut für Programmstrukturen und Datenorganisation, Universität Karlsruhe (TH) 2007

30. *Michael Reimann*: Simulationsmodelle im Verkehr. Universität Karlsruhe (TH) 2007

31. *Roland Chrobok, Sigurour F. Hafstein, Andreas Pottmeier*: OLSIM: A New Generation of Traffic Information Systems. Universität Duisburg-Essen 2003

32. *Larry E. Owen, Yunlong Zhang, Lei Rao, Gene McHale*: Traffic Flow Simulation Using CORSIM. Proceedings of the 2000 Winter Simulation Conference, Austin, USA 2000

33. *Eckehard Schnieder*: Verkehrsleittechnik: Automatisierung des Straßen- und Schienenverkehrs. Springer Verlag, Berlin 2007

34. *Kai Erlemann*: Objektorientierte mikroskopische Verkehrsflusssimulation. Fakultät für Bauingenieurwesen, Ruhr-Universität Bochum 2007

35. *Michael Schreckenberg, Andreas Schadschneider, Kai Nagel*: Zellularautomaten simulieren Straßenverkehr. VCH-Verlagsgesellschaft mbH, Weinheim 1996

36. *M. Bando, K. Hasebe, A. Nakayama, A. Shibata, Y. Sugiyama*: Dynamical model of traffic congestion and numerical simulation. The American Physical Society 195

37. *Nils Gustaf Eissfeldt*: Vehicle-based modelling of traffic - Theory and application to environmental impact modelling. Mathematisch-Naturwissenschaftliche Fakultät, Universität Köln 2004

38. *Andreas Schadscheider*: Physik des Straßenverkehrs. Institut für Theoretische Physik, Universität Köln 2004

39. VISSIM Produktbeschreibung. http://www.ptvag.com/index.php?id=1801, geprüft:

31.08.2009

40. *Rainer Wiedemann*: Simulation des Straßenverkehrsflusses. Universität Karlsruhe 1974

41. *Martin Fellendorf, Peter Vortisch*: Validation of the Microscopic Traffic Flow Model VISSIM in Different Real-World Situations. Transportation Research Board 2001

42. *Loren Bloomberg, Jim Dale*: A Comparison of the VISSIM and CORSIM Traffic Simulation Models On A Congested Network. Transportation Research Record 2000

43. *Daniel Krajzewicz, Michael Bonert, Peter Wagner*: The Open Source Traffic Simulation Package SUMO. German Aerospace Centre, Institute of Transportation Research, Berlin 2006

44. *Elmar Brockfeld, Reinhart Kühne, Peter Wagner*: Calibration and Validation of Microscopic Traffic Flow Models. Transportation Research Board 2005

45. SUMO Projektdokumentation. http://sourceforge.net/apps/mediawiki/sumo/index.php?title=Main_Page, geprüft: 31.08.2009

46. *Michael Behrisch, Rüdiger Ebendt, Daniel Krajzewicz, Yun-PangWang*: Verkehrssimulation und Optimierung. Institut für Verkehrssystemtechnik, Deutsches Zentrum für Luft- und Raumfahrt e.V. 2008

47. *Christopher Schaumann*: Entwurf und prototypische Implementierung einer flexiblen SOTIS Architektur. TU Dortmund 2009

48. OSM Protocol Version 0.6. http://wiki.openstreetmap.org/wiki/OSM_Protocol_Version_0.6, geprüft: 31.08.2009

49. *Lars Wischof, André Ebner, Hermann Rohling, Matthias Lott, Rüdiger Halfmann*: Self-Organizing Traffic Information System. Universitätsverlag Karlsruhe 2005

50. *Johann Nowicki*: Marktentwicklung für Kfz-Navigation - ADAC-Schätzung und Prognose bis 2005. ADAC - Verkehrspolitik und Verbraucherschutz 2002

51. Der moderne Softwareentwicklungsprozess mit UML. http://www.highscore.de/uml/, geprüft: 31.08.2009

52. *Stephan Eichler, Christoph Schroth, Timo Kosch, Markus Strassberger*: Strategies for Context-Adaptive Message Dissemination in Vehicular Ad Hoc Networks. IEEE Computer Society, , San Jose, USA 2006

53. *Alfons Kemper, André Eickler*: Datenbanksysteme - 6. Auflage. Oldenbourg Verlag 2006

54. *André Ebner, Hermann Rohling, Lars Wischof*: Self-Organizing Traffic Information

System based on Car-to-Car Communication: Prototype Implementation. 1st International Workshop on Intelligent Transportation, Hamburg 2004

55. *Matthias Gerlach*: VaneSe - An approach to VANET security. Proceedings of V2VCOM 2005, San Diego, USA 2005

56. *Matthias Gerlach*: Assessing and Improving Privacy in VANETs. Fraunhofer-Institut für Offene Kommunikationssysteme, Berlin 2006

57. *Falko Dressler, Tobias Gansen, Christoph Sommer and Lars Wischhof*: Requirements and Objectives for Secure Traffic Information Systems. Proceedings of 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, Atlanta, USA 2008

58. *Krishna Sampigethaya, Leping Huang, Mingyan Li, Radha Poovendran, K. Matsuura, K. Sezaki*: CARAVAN: Providing Location Privacy for VANET. Proceedings of Embedded Security in Cars (ESCAR), Köln 2005

59. *Erich Gamma, Richard Helm, Ralph Johnson*: Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software. Addison-Wesley, München 2004

60. *Heide Balzert*: Lehrbuch der Objektmodellierung: Analyse und Entwurf mit der UML 2. Spektrum Akademischer Verlag, München 2005

61. eWorld Projektdokumentation. http://eworld.sourceforge.net/, geprüft: 31.08.2009

Anhang

A Benutzerhandbuch

A.1 Konfiguration von SUMO

SUMO ist Open Source und unter *http://sumo.sourceforge.net/* im Internet verfügbar. In dieser Arbeit wurde die Version 0.11.0 verwendet. Zur Nutzung in der Simulationsumgebung muss das verwendete Kartenmaterial im SUMO-Format und eine Datei mit der Spezifikation der zu simulierenden Fahrzeuge vorliegen. Diese Parameter werden über eine Konfigurationsdatei an SUMO übergeben. Nachfolgend sind die jeweils notwendigen Befehle sowie eine Beispiel-Konfigurationsdatei aufgeführt.

Nutzung von OpenStreetMap-Kartenmaterial[33]

- Abruf des Kartenmaterials über die OSM-API:
  `wget.exe "http://api.openstreetmap.org/api/0.6/map?bbox=13.51,52.42,13.555,52.4385" -O my_osm_net.osm.xml`
- Konvertierung in OSM-Format:
  `netconvert --osm my_osm_net.osm.xml -o my_sumo_net.net.xml --use-projection`

Erzeugung zufälliger Routen für Fahrzeuge[34]

- Der Parameter *R* bestimmt, wie viele Fahrzeuge pro Sekunde im Intervall von *b* bis *e* erzeugt werden sollen:
  `duarouter –net=my_sumo_net.net.xml -R <ANZAHL> --output-file=my_routes.rou.xml -b <START> -e <STOP>`

33 Mit Hilfe von *wget* zum Aufruf der OSM-API, siehe *http://wget.addictivecode.org/*.

34 Für weitere Erläuterungen siehe *http://sumo.sourceforge.net/docs/gen/user_chp05.shtml#user_chp05-own_routes-random*. Hier ist auch der Import von existierenden Routen aus anderen Quellen beschrieben.

SUMO-Konfigurationsdatei

Die Konfigurationsdatei enthält hauptsächlich die Verweise auf die Dateien mit dem Straßennetz und den Fahrzeugdefinitionen.[35]

```
<configuration>

    <input
        net-file="my_sumo_net.net.xml"
        route-files="my_routes.rou.xml"
    />

    <output
        vehroute-output="vehroutes.xml"
    />

    <report
        no-duration-log="true"
        no-step-log="true"
    />

</configuration>
```

A.2 Konfiguration von KSS

Der implementierte Prototyp ist Open Source und unter *https://sourceforge.net/projects/ossotis/* im Internet verfügbar. Die Ausführung erfolgt über die Klasse *Simulation*. Drei Parameter werden benötigt: die Funkreichweite *r*, die Funkbandbreite *b* und der OSSOTIS-Verbreitungsgrad *v* (siehe Abschnitt 5.2.4). Die Simulation wird nach jedem Schritt unterbrochen und kann mit *[ENTER]* fortgeführt werden, die Eingabe von *exit* beendet die Simulation. Die Ausgabe der Ergebnisse erfolgt über die Kommandozeile.

35 Die weiteren Parameter sind unter *http://sumo.sourceforge.net/docs/gen/user_chp08.shtml#user_chp08-configs* beschrieben.

B Klassendiagramme

B.1 Simulationsumgebung

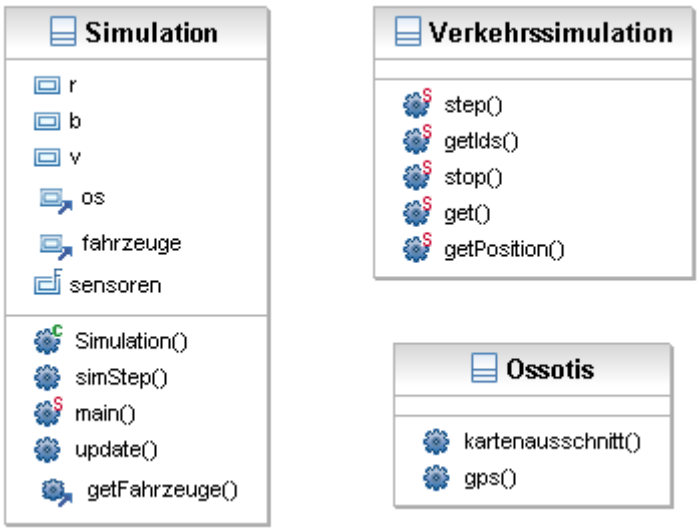

Abbildung 20: Klassendiagramm Simulationsumgebung

B.2 KoSit-Komponente

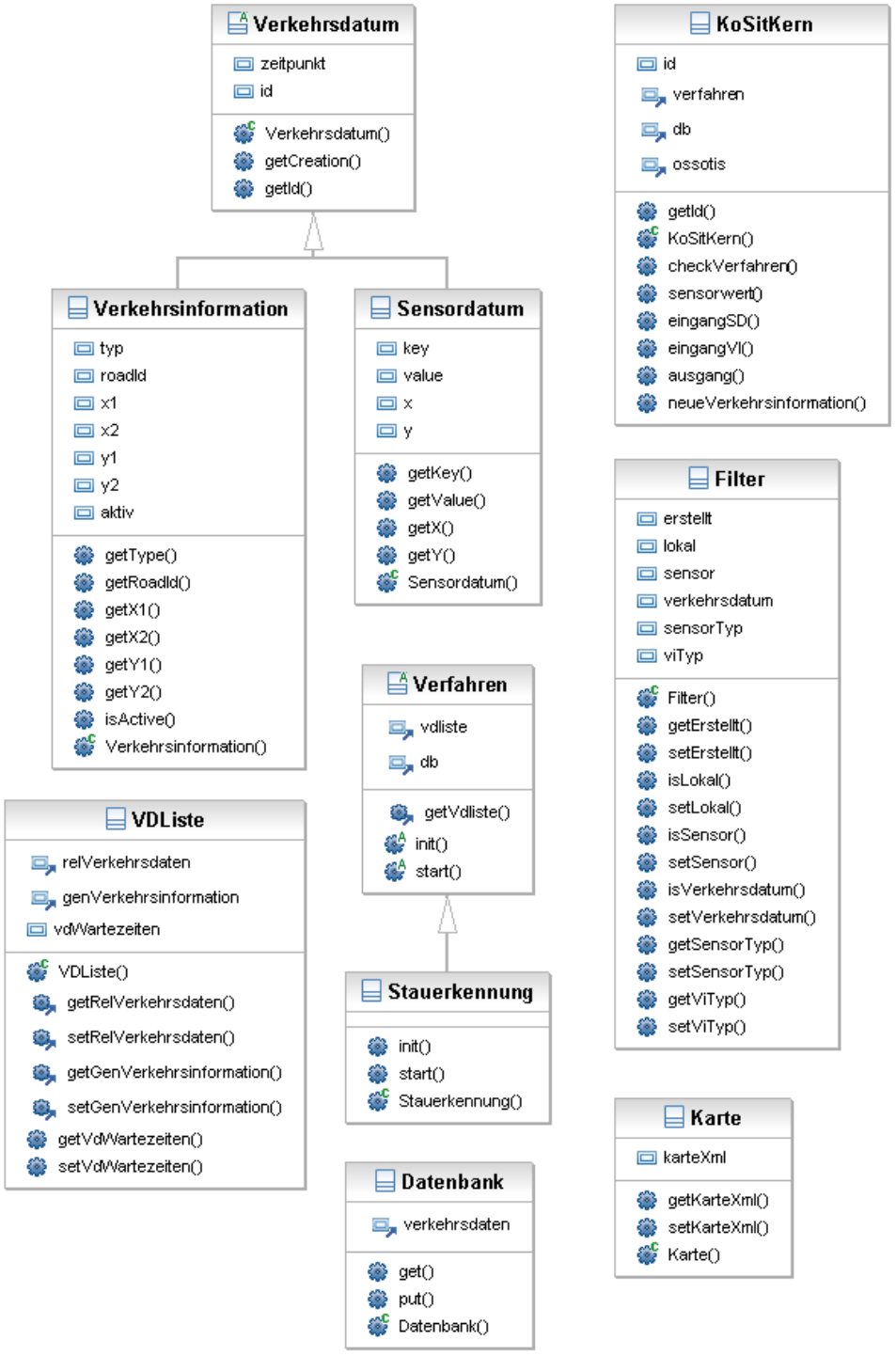

Abbildung 21: Klassendiagramm KoSit-Komponente