Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Grundseminar Ausarbeitung

Timo Häckel

Schnittstellen und Interaktionen zwischen fahrer- und fahrzeugbezogenen Diensten

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Timo Häckel

Schnittstellen und Interaktionen zwischen fahrer- und fahrzeugbezogenen Diensten

Grundseminar Ausarbeitung eingereicht im Rahmen der Grundseminar Abschlussprüfung

im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Franz Korf

Eingereicht am: 14. Juli 2016

Timo Häckel

Thema der Arbeit
Schnittstellen und Interaktionen zwischen fahrer- und fahrzeugbezogenen Diensten

Stichworte
Systemarchitektur, Automobile, Dienste, Schnittstellen, Interaktionen

Kurzzusammenfassung
Die großen Internet-Unternehmen versuchen das Auto in das Internet of Things zu integrieren. Dies birgt für die Automobilhersteller viele Probleme, die gelöst werden müssen. Unter anderem muss, um diese Integration umzusetzen, eine gut strukturierte Dienste-Architektur aufgebaut werden. In diesem Artikel werden Rechercheergebnisse aus dem Themenbereich Automobil und Dienste gesammelt und in Verbindung gebracht, aber auch die Grundlegenden Quellen wie Konferenzen und damit der aktuelle Stand der Forschung beschrieben.

Timo Häckel

Title of the paper
Interfaces and Interactions for Passenger and Driving related Services

Keywords
System architecture, automotives, services, interfaces, interactions

Abstract
The big players of the Internet try hard to integrate cars into the Internet of Things. This creates many challenges to solve for automotive manufacturers. Amongst other things, a well structured service architecture is needed to transpose this integration. This article contains results of research in the topic automotives and services, as well as sources for basic information like conferences. Another goal is to provide the current state of the art in research for this topic.

Inhaltsverzeichnis

1 Einleitung

Die Automobilindustrie befindet sich in einem starken Wandel und die Software Entwicklung nimmt eine immer größere Rolle ein. Außerdem wird das Auto immer stärker mit dem Internet verknüpft und somit Teil des Internets der Dinge. Hierbei spielen Cloud-Anwendungen eine große Rolle, durch die automatisiertes Fahren und andere Fahrerassistenzsysteme ermöglicht werden sollen.[35] Um die Menge an Software zu beherrschen, die durch die große Nachfrage an Funktionen durch die Anwender entsteht, und die Kommunikation mit anderen *Dingen* zu ermöglichen, soll eine neue Systemarchitektur entwickelt werden, die die Kommunikation und Interaktion der verschiedenen Komponenten im Auto erleichtern soll.
In dieser Arbeit werden die im Rahmen des Grundseminars gesammelten Rechercheergebnisse für ein solches Vorhaben diskutiert, sowie das Themengebiet der Softwareentwicklung im Automobilbereich beleuchtet. Sie wurde im CoRE Projekt der HAW-Hamburg durchgeführt, welches von Prof. Dr. Franz Korf geleitet wird. Dieses Projekt befasst sich mit der Weiterentwicklung der Kommunikation zwischen Komponenten in Automobilen. [12]

1.1 Softwareentwicklung für Automobile

Die Softwareentwicklung im Automobilbereich nimmt einen immer größeren Teil in der Weiterentwicklung von Automobilen ein. Die Komplexität der Aufgaben steigt und die Nutzer verlangen immer mehr Funktionalität. Ein weiterer Grund für diesen Verlauf, sind die sinkenden Hardwarekosten von Recheneinheiten und Steuergeräten. Abbildung 1.1 zeigt die Entwicklung der Bedeutung von Software im Fahrzeug, sowohl in der Softwarekomplexität, als auch im Anteil am Gesamtwert des Automobils. Die Aufgaben der entwickelten Software reichen von Verbesserungen im Komfort, über Steigerung der Performance, bis zur Verstärkung der Sicherheit. Die Abbildung zeigt sehr deutlich, wie stark sich die Automobilindustrie in der Zukunft auf Software fokussieren wird, um dem Käufer neue Funktionen zu bieten.[38, 25]
Zur Umsetzung neuer Funktionen, werden derzeit sehr viele getrennte Steuergeräte eingesetzt und es wird versucht, die sicherheitsrelevanten Systeme von den Komfortsystemen fernzuhalten. Diese beiden Bereiche verschwimmen in der Zukunft jedoch stark. Wenn man an automatisiertes Fahren denkt, stellt man sehr schnell fest, dass dieses System mit allen anderen im Auto (und auch außerhalb des Autos) existierenden Funktionen zusammenhängt, von Bremse über Lenkung bis zum Navigationssystem und einer Internetverbindung für aktuelle Meldungen.[29] Insbesondere Cloudsysteme werden in der Zukunft genutzt werden, um die Datenmengen zu analysieren, die das automatisierte Fahren mit sich bringen wird.[35] Um für solche Systeme gewappnet zu sein, wird eine neue Systemarchitektur benötigt. Dadurch könnte auch eine Vielzahl von Steuergeräten abgebaut werden und sehr viel Herstellungskosten eingespart werden.

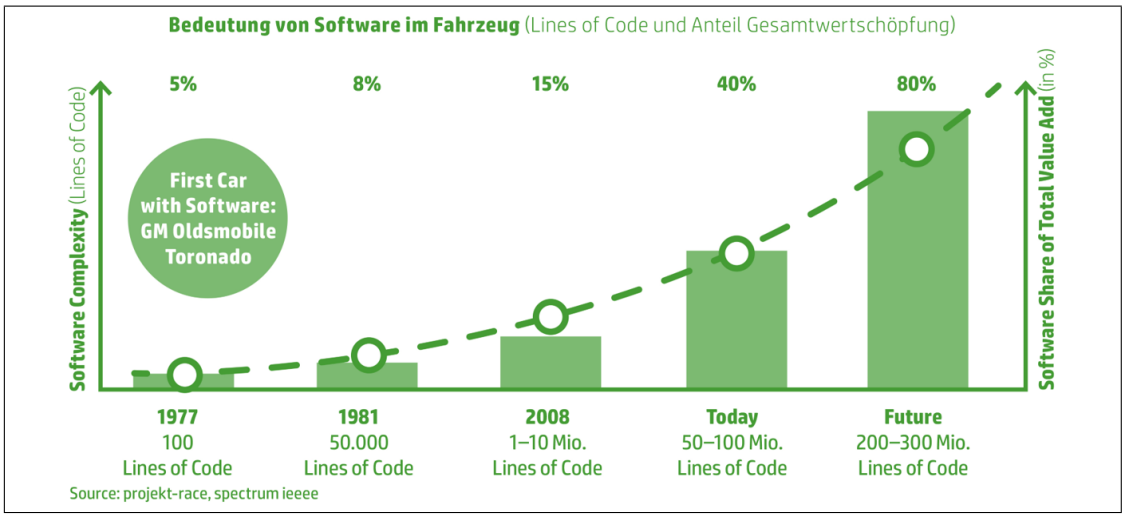

Abbildung 1.1: Bedeutung der Software im Fahrzeug in der Entwicklung von 1977 bis in die Zukunft. [25]

1.2 Überblick

Das langfristige Ziel dieses Projektes ist es, eine Dienstarchitektur zu entwickeln, um Schnittstellen und Interaktionen zwischen fahrer- und fahrzeugbezogenen Diensten zu schaffen. Dabei sind vor allen Dingen die vielen Features und unterschiedlichen Entwicklungszyklen von Diensten eine große Herausforderung. Wenn das Auto in das Internet der Dinge integriert wird, treffen zwei verschiedene Welten aufeinander. Auf der einen Seite, die schnelllebigen Dienste des Internets mit kurzen Entwicklungszyklen und auf der anderen Seite, die Dienste des Automobils mit Entwicklungszyklen von mehreren Jahren und einer Produktlebensdauer von mehreren Jahrzehnten. [31, 33] Das Ziel dieser Ausarbeitung besteht darin, sich einen Überblick über die verschiedenen Aspekte dieses Themengebietes zu verschaffen.
Im Folgenden werden die verschiedenen Themen meiner Recherche in den Bereichen Dienstarchitekturen (Kapitel 2), Schnittstellen und Interaktionen (Kapitel 3), aber auch im Bereich der Softwareentwicklung für Automobile besprochen. Hierbei wird der State of the Art in Forschung und Industrie beleuchtet (Kapitel 4). Abschließend werden die Ergebnisse gesammelt, sowie die wichtigsten Experten und Konferenzen vorgestellt und ein Ausblick für die Weiterentwicklung des Projekts gegeben (Kapitel 5).

2 Dienste

Um über eine Architekturentwicklung für sichere Schnittstellen und Kommunikationen zwischen Diensten zu sprechen, sollte als erstes geklärt werden, was Dienste überhaupt sind. Douglas K. Barry beschreibt Dienste in seinem Buch[26] als Software- und/oder Hardware-Einheit in einem komplexen System. Diese Einheit stellt eine eigenständige Business Funktion zur Verfügung, wobei das Ziel bei der Entwicklung die Wiederverwendbarkeit dieser Funktion ist.
Laut Barray gibt es zwei Arten von Diensten: Atomare und Kompositionen. Diese werden, frei aus dem Englischen übersetzt, wie folgt definiert:

> *Ein atomarer Dienst ist eine wohl definierte, abgeschlossene Funktion, die nicht von dem Kontext oder Zustand anderer Dienste abhängt. Eine Komposition von Diensten ist eine Zusammenstellung von atomaren Diensten oder weiteren Dienstkompositionen. Ein Dienst in einer solchen Komposition kann vom Kontext oder Zustand anderer Dienste abhängen, die in derselben Dienstkomposition enthalten sind.[26]*

Im Folgenden werden die Anforderungen der Automobilindustrie an solche Dienste beschrieben, sowie die Dienste in verschiedene Kategorien eingeteilt.

2.1 Anforderungen an Dienste

Die Anforderungen an Dienste in einem Automobil ergeben sich aus den Anforderungen an das Auto selbst: Hohe Zuverlässigkeit und Verfügbarkeit, lange Produktlebensdauer, sowie Produkt- und Applikationsvielfalt. Wenn man diese auf die Softwareentwicklung von Diensten überträgt kann man folgende Anforderungen definieren: Redundanz und Qualität, um die Zuverlässigkeit und Verfügbarkeit zu gewährleisten; Migrierbarkeit und Legacy, um eine Portierung auf weitere Modelle, sowie eine lange Produktlebensdauer zu ermöglichen; Offenheit mit Standard Interfaces, um die Produkt- und Applikationsvielfalt, sowie die Systemkomplexität beherrschbar zu halten.[33, 30]
Es ist jedoch wichtig, bei den Anforderungen Unterscheidungen der verschiedenen Domains für Automobilsoftware zu machen. Üblicherweise wird diese Software in fünf verschiedene Domains eingeteilt: Mutlimedia (MMI) Software, Mensch/Komfort Software, Software für Sicherheitselektronik, Antrieb- und Fahrwerksteuerung und Infrastruktur Software (wie zum Beispiel Diagnose Software). In der heutigen Zeit kommt jedoch noch ein sechster Bereich dazu, welcher als Cloudsoftware bezeichnet werden könnte.[31, 30, 36]

2.2 Dienstkategorien

Aufgrund der verschiedenen Domains und der damit verbundenen Vielfalt an Diensten, sind die Anforderungen nicht für alle Dienste gleich. Wenn man die Anforderungen eines Bremssystems mit denen eines Entertainment Systems vergleicht, kann man deutliche Unterschiede

in den verschiedenen Anforderungsbereichen sehen. Daher unterscheiden wir fahrer- und fahrzeugbezogene Dienste. Fahrerbezogene Dienste sind solche, die keinen spezifischen Mehrwert für die Nutzung des Fahrzeugs haben und sich auf den Fahrer beziehen, wie zum Beispiel das Entertainment System. Fahrzeugbezogene Dienste sind im Gegensatz dazu solche, die die Nutzung des Fahrzeugs erweitern und sich auf das Fahrzeug an sich beziehen, wie zum Beispiel ein Abstandshalteassistent. Beide dieser Kategorien sind nicht an die Grenzen des Fahrzeugs gebunden. So wäre Beispielsweise ein Clouddienst für automatisiertes Fahren, immer noch ein fahrzeugbezogener Dienst.[33, 36]
In Abbildung 2.1 sind die beiden wichtigsten Eigenschaften in Automobilen (Sicherheit und Echtzeit) in Zusammenhang mit den verschiedenen Dienstkategorien gebracht. So sind die Sicherheits- und Echtzeitanforderungen bei fahrerbezogenen Diensten eher niedrig, bei fahrzeugbezogenen jedoch eher hoch. Dazwischen befinden sich verschiedene Übergangsbereiche, die entweder sehr hohe Sicherheitsanforderungen oder sehr hohe Echtzeitanforderungen haben. Während zum Beispiel ein Softwareupdate nur sehr geringe Echtzeitanforderungen hat, hat es auf der anderen Seite sehr hohe Sicherheitsanforderungen, da Fehler in einer Katastrophe enden können. Genau umgekehrt verhält es sich bei den Interaktionen zwischen Mensch und Auto. Aufgrund der begrenzten und fest definierten Möglichkeiten zur Interaktion gibt es nur sehr geringe Sicherheitsanforderungen, allerdings sehr hohe Anforderungen an die Echtzeit, da auf die Nutzereingaben wie Bremsen oder Lenken im Auto sofort reagiert werden muss. In der Mitte dazwischen sind die Übergangsbereiche, in denen beides durchschnittlich wichtig ist. Dazu zählt zum Beispiel der Datenaustausch.[33, 14]
Man kann also sagen, dass die Einteilung in die beiden Kategorien fahrer- und fahrzeugbezogene Dienste aufgrund der unterschiedlichen Anforderungen sinnvoll ist.

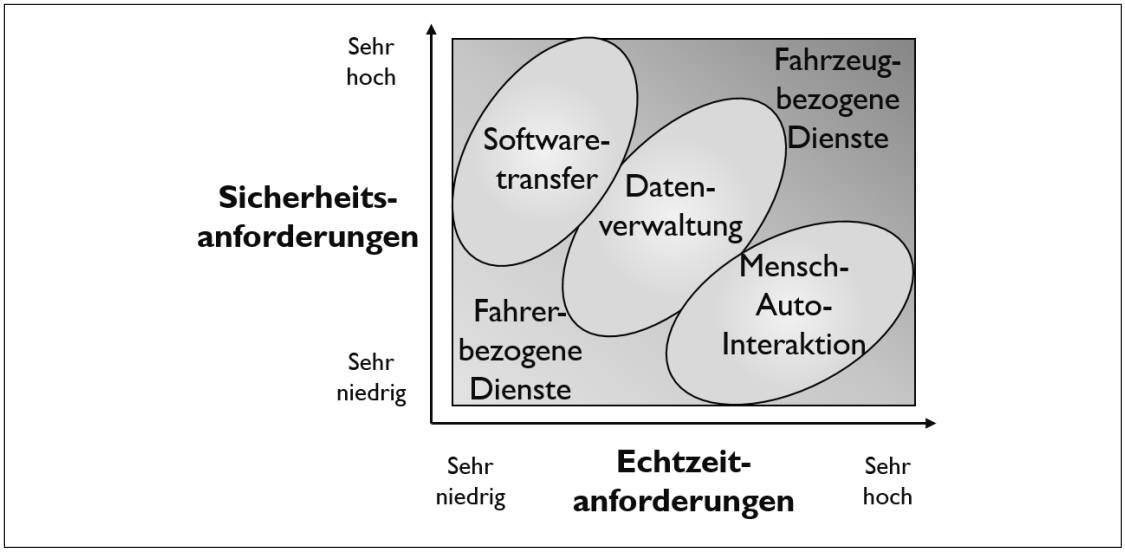

Abbildung 2.1: Die verschiedenen Dienstkategorien und ihrer Bedeutungen im Automobil nach [33].

3 Schnittstellen und Interaktionen

Schnittstellen werden benötigt, damit Dienste auf andere Dienste zugreifen und ihnen ihre Funktionen anbieten können. Außerdem kann mit ihnen definiert werden, wie auf eine Komponente zugegriffen werden darf und wer Zugriff hat.[27]
Interaktionen sind festgelegte Kommunikationsmuster zum Austausch von Nachrichten, die meist über eine Middleware erfolgen. Sie definieren die Nutzung der Schnittstellen.[27]
Diese Schnittstellen werden meist in einer klassischen Service orientierten Architektur[37] aufgebaut. Diese Architekturen sind in vielen IT-Bereichen bereits bekannt und etabliert. Wie man diese Konzepte allerdings auf das Auto übertragen kann, ist noch nicht geklärt. Diese Frage soll im Grundprojekt (siehe Ausblick 5.2) geklärt werden. Im Folgenden werden mit Web Services und OSGi zwei verschiedene Möglichkeiten zur Definition dieser Schnittstellen vorgestellt, um die etablierten Konzepte als Vorlage für dieses Projekt darzulegen.

3.1 Webbasierte Dienstarchitekturen

Das Web ist eine sehr gute Vorlage für Dienstarchitekturen, da diese dort schon seit vielen Jahren von einer großen Entwicklergemeinde in Millionen von Web Services genutzt werden. Außerdem nutzen diese Web Services die gleiche Infrastruktur, wie die Real-Time Ethernet Varianten, die das zukünftige Basisnetz der Kommunikation in Automobilen werden. Dadurch könnten die verschiedenen Schichten der Internetarchitektur ebenfalls übernommen werden, um die Kommunikation sicher zu gestalten.[12] Der wichtigste Punkt ist jedoch, dass das Internet bereits klare Standards hat. Diese Standards werden von der W3C weiterentwickelt und verwaltet. Sie definieren, wie ein Dienst aufgebaut wird und wie die Kommunikation zwischen Diensten abzulaufen hat.[24] Dieser Aspekt ist besonders interessant, da das Auto durch das IoT auch externe Dienste beziehen wird und somit diese Standards erfüllen sollte.
Ein wichtiger Standard ist in diesem Zusammenhang die **W**eb **S**ervices **D**escription **L**anguage (WSDL), welche als XML[10] basierte Beschreibungssprache für Web Services dient. In ihr wird definiert, wie ein Klient mit einem Dienst interagieren muss. Es wäre vorstellbar, dass in einer solchen Beschreibungssprache, auch Dienste in Automobilen sehr gut beschreibbar sind, wobei dies den Einsatz einer Middleware als Vermittler zur Folge hätte.[23, 27, 26]
Ein weiterer nützlicher Standard ist das **Re**presentational **S**tate **T**ransfer (ReST) Protokoll, zur Kommunikation zwischen Web Services. Der Fokus dieses Protokolls liegt vor allen Dingen auf der Anforderung von Ressourcen. Diese werden von einem Klienten (*Service Consumer*) bei einem Dienst (*Service Provider*) mittels einer ReST Nachricht angefragt und als XML zurückgeschickt.[27, 26] Diese Art von Kommunikation ermöglicht die Reduzierung von Diensten auf die Nutzung von Ressourcen, was auch ein vorstellbares Verhalten im Auto ist.

3.2 OSGi

Eine alternative zu der Dienstarchitektur des Webs, bietet die **O**pen **S**ervices **G**ateway **i**nitiative (OSGi) Alliance[17]. Diese ist ein Industriekonsortium, welches das OSGi Framework spezifiziert. OSGi ist ein Java basiertes Framework, dass als Middleware für Hardwareunabhängige dynamische Softwareplattformen genutzt wird. Es verwaltet die Dienste eines Systems in einer Service Registry und ermöglicht die Kommunikation und Interaktion zwischen diesen Diensten.[17, 18]
In Abbildung 3.1 ist der Aufbau des OSGi Framework dargestellt. Die Grundlage des Systems ist, auf den beiden untersten Ebenen, das Betriebssystem und die darin laufende Java VM. Darauf liegt die *Execution Environment*, welche die Klassen und Methoden von OSGi definiert. Darüber liegen drei Schichten, die die Verwaltung der Dienste (in OSGi als *Bundles* definiert) übernehmen. Die *Modules* Ebene definiert, wie neue Bundles dynamisch importiert oder exportiert werden können. Der *Life Cycle* bietet eine API zum Installieren, Starten, Stoppen, Updaten und Deinstallieren von Bundles. Ganz oben im OSGi Framework liegt die *Services* Schicht, welche als Schnittstelle für Bundles fungiert. Bei ihr können mithilfe eines *publish-find-bind* Modells[37] Java Objekte registriert oder gefunden werden. Die auf der linken Seite dargestellten *Bundles*, sind die gegen das OSGi Framework entwickelten Dienste eines Entwicklers. Durch das *Security* Modul können Sicherheitsregularien festgelegt werden.[17, 18, 34]
Insgesamt kann man sagen, dass sich OSGi sehr gut für eine Ein-Rechner-Architektur eignet und daher, bei einer Zentralisierung, auch im Auto genutzt werden könnte.

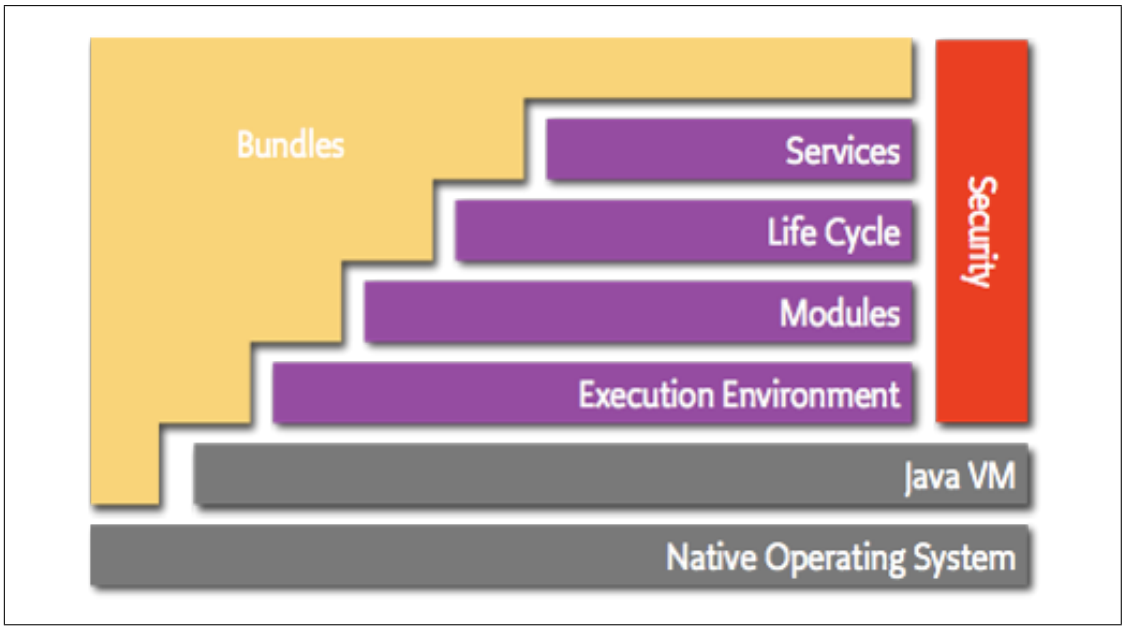

Abbildung 3.1: Aufbau des OSGi Frameworks.[17]

4 Industrie und Forschung

In diesem Kapitel werden verschiedene Projekte der Industrie und Forschung vorgestellt. Diese werden insbesondere mit den Eigenschaften einer Dienstarchitektur verglichen und auf Erkenntnisse in der Kommunikation zwischen diesen Diensten untersucht.
In der Industrie beschäftigen sich viele verschiedene Projekte mit der Integration des Autos in das Internet der Dinge. Selbstverständlich entwickelt die Automobilindustrie, repräsentiert durch die großen Autohersteller, an dieser Integration und arbeitet auf die Nutzung von Diensten hin. Aber auch die Vertreter des Internets der Dinge möchten das Auto integrieren. Besonders Apple und Google mit den Betriebssystemen Android und IOS, allerdings auch mit anderen Projekten arbeiten in diesem Bereich. Außerdem ist die IAV als Dienstleister mit der Verbindung zwischen IT und Automotive im Internet der Dinge beschäftigt. Neben der Industrie gibt es auch noch einige Forschungsprojekte an Forschungsinstituten, aber auch an Hochschulen und Universitäten. Im Folgenden werden diese vier Perspektiven erörtert.

4.1 Autohersteller

Die Autohersteller arbeiten stark daran, das Auto in das Internet der Dinge zu Integrieren. Dabei steht allerdings stets das Auto im Mittelpunkt. Systeme wie Volkswagen Car Net[6], BMW Connected Drive[4], Mercedes me connect[16], aber auch von allen anderen Autoherstellern, bieten Möglichkeiten zum entfernten Entriegeln des Fahrzeuges, mitnehmen von Navigations- und Entertainmentdaten und sogar eigene App Stores im Auto.
Es sind leider keine Entwicklungsdetails der Autohersteller öffentlich, die Aufschluss über Architekturänderungen für solche Konzepte geben könnten. Diese Konzepte gehen allerdings nicht so weit, dass sie fahrerbezogenen Dienste mit fahrzeugbezogenen Diensten kommunizieren lassen würden. Diese beiden Systeme arbeiten vollkommen unabhängig voneinander und bieten maximal Lesezugriff. Auch dort existiert also großes Potential zur Weiterentwicklung durch eine neue Systemarchitektur.

4.2 Unternehmen aus dem IoT

Google und Apple sind, als Entwickler der größten Betriebssysteme im Mobilbereich, unter den wichtigsten Vertretern des Internets der Dinge. Beide Konzerne arbeiten daran, das Auto in das Internet der Dinge zu integrieren und vor allem, durch eine Smartphone Integration im Auto, das Internet der Dinge ins Auto zu bringen. Beide Hersteller haben mit den Produkten *Android Auto*[2] und *Apple Carplay*[7] Plattformen dafür geschaffen, die bereits in den neuen Modellen des Jahres 2016/2017 in fast allen Autos verfügbar seien werden.
Android Auto und Apple Carplay bieten sehr ähnliche Funktionen. Wenn sich das Smartphone mit dem Auto verbindet, wird der Bildschirm des Smartphones auf das Multimedia-System des Autos umgeschaltet. Beide Unternehmen haben neue, speziell ans Auto angepasste Interfaces

entwickelt, die das Nutzen der Apps des Smartphones im Auto ermöglichen. Sie bieten Navigationsdienste, Nachrichten- und Telefondienste, sowie Multimediadienste. Außerdem können Entwickler auf der ganzen Welt weitere Dienste für diese Plattform entwickeln.[2, 7]
Das Internet der Dinge hat also bereits Einzug in das Auto erhalten. Allerdings bleibt auch hier anzumerken, dass das Auto lediglich als Ein- und Ausgabegerät verwendet wird und das Smartphone keinerlei Zugriff auf die fahrzeuginternen Dienste erhält. In unserem Projekt wollen wir noch etwas weitergehen und externen Diensten (z.B. Cloud Diensten) ermöglichen, auch die Auto internen Dienste ansteuern zu können.

Abgesehen von Smartphones nimmt Google mit seinem *Self-Driving Car Project* an der Weiterentwicklung der Automobile teil. Und auch über ein Apple *iCar* gibt es einige Gerüchte. Diese werden vor allen Dingen dadurch geschürt, dass Apple seit einiger Zeit viele Experten für Elektroautos einstellt. Allerdings wurde keines dieser Gerüchte von Apple bestätigt.
Im Gegensatz zu den etablierten Autoherstellern, hat das Google Car allerdings einige Besonderheiten. Während bei BMW, VW, Mercedes, usw. der Spaß am Fahren und die Weiterentwicklung des Autos im Vordergrund steht, will Google ein reines Transportmittel entwickeln. Dieses könnte eher einen Taxiersatz darstellen, der Personen auf Knopfdruck zu einem gewünschten Ziel transportiert. [11] Dabei wäre die Kommunikation zwischen Diensten fest definiert, da keine Nutzerinteraktion mehr stattfinden muss. Somit ist die Dienstarchitektur, entgegen dem Ansatz unseres Projekts, nach außen abgeschlossen.

4.3 Dienstleister

In der Automobilindustrie gibt es verschiedene Dienstleistungsunternehmen. Die Besonderheit dieser Unternehmen ist, dass sie keine eigenen Produkte auf dem Markt vertreiben. Sie bieten lediglich eine oder mehrere Dienstleistungen an. Als Beispiel soll hier die Ingenieursgesellschaft Auto und Verkehr (IAV) dienen.
Die IAV ist ein Dienstleistungsunternehmen, welches Entwicklungen im Automobilbereich durchführt. Vom Blaulicht von der Polizei, bis zu einer Smartphone App, entwickeln sie je nach Auftrag. Laut IAV zählen nahezu alle Automobilhersteller und ihre Systemzulieferer zu ihren Kunden. [14]
Neben ihrer Entwicklungsdienstleistung besitzt die IAV allerdings auch eine eigene Forschungsabteilung, welche neue innovative Konzepte überprüft. Ein gemeinsames Forschungsfeld ist die Integration des Autos in das Internet der Dinge. Dabei geht es sowohl darum das Auto als Ding in die Umwelt zu integrieren, als auch die Außenwelt in das Auto hinein zu lassen. Hierfür ist eine neue Systemarchitektur unabdingbar. Außerdem unterstützt die IAV die CoRE Group der HAW Hamburg[12] und wird diese auch bei dem Projekt (Kommunikation und Interaktion zwischen fahrer- und fahrzeugbezogenen Diensten), als Partner unterstützen.

4.4 Forschung

In der Forschung beschäftigen sich viele Projekte mit den Teilbereichen der Automobilindustrie. Beispiele dafür sind das Institut für Automobil Forschung (IAF)[19] und der Verband der Automobilindustrie (VDA) mit ihrer Forschungsvereinigung Automobiltechnik (FAT)[20]. Neben diesen Forschungsgruppen gibt es auch einige Hochschulen[12, 3] und Universitäten[32], die im Automobilbereich forschen. Der Schwerpunkt dieser Projekte liegt dabei meist auf Energieeffizienz. Besonders interessant ist daher das Visio.M Projekt der Technischen Universität München, da es sich ebenfalls mit dem Aufbau einer Dienstarchitektur befasst. Dieses wird im Folgenden beschrieben.

Der Visio.M ist der Prototyp eines umweltfreundlichen, sparsamen und sicheren Elektroautos und wurde an der Technischen Universität München entwickelt. Das Projekt wurde in Kooperation mit führenden Konzernen der Automobilindustrie, wie zum Beispiel BMW, Daimler, Continental und vielen mehr durchgeführt. Um die vielen Fachbereiche einer Universität auszunutzen wurden alle Aspekte, wie Design, Vermarktung, Fahrzeugtechnik und Informatik, an den verschiedenen Departments der TU umgesetzt.[22]
Zur Informationstechnischen Umsetzung wurde der Automotive Service Bus als Nachrichtenkanal entwickelt. Dies ist ein OSGi (siehe Kapitel 3.2) basiertes Dienste System, mit dessen Hilfe die verschiedenen Komponenten Nachrichten austauschen können. Als Einschränkung wird hierbei auf Fahrzeugdaten nur lesender Zugriff gewährt. Nur definierte zuvor festgelegte Komponenten erhalten von einer zentralen Steuereinheit Schreibrechte für diese Daten. Auf der Hardwareseite wird dieses System durch einen zentralen Linux Board Computer verwaltet.[22, 32]
Da der Visio.M ein Open Source Projekt ist, sind auch die Architekturdetails öffentlich. Der Aufbau der IT-Architektur (Abbildung 1 im Anhang 1) des Visio.M besteht aus vier Bereichen, die durch ein zentrales Steuergerät verbunden sind. Auffällig sind die drei verschiedenen CAN-Bus Netze[39, 28] *Antiebs-CAN*, *Batterie-CAN* und *Sicherheits-CAN*. Diese sollen durch das dazwischenliegende Steuergerät, sowohl von einander, als auch von der Außenwelt abgeschirmt werden, um Angriffe zu verhindern. Das Steuergerät bietet einen Lesezugriff auf nahezu alle Daten. Außerdem ist das zentrale Steuergerät über einen Ethernet Switch mit der darüber liegenden Ebene verbunden. Diese ist als eine Dienstarchitektur aufgebaut, die durch einen *Web-PC* und ein I-Pad realisiert wurde.[32] Um nicht nur Lesezugriff sondern auch Schreibzugriff zu erlauben, müsste die Kommunikation deutlich besser abgesichert werden. Dieser Schreibzugriff ist ein Aspekt, der weiter Forschung und Entwicklung erfordert.

5 Fazit

Es existieren bereits verschiedene Ansätze, um das Auto in das Internet der Dinge zu integrieren. Bisher reicht diese Integration allerdings nur bis zum Rand des Autos, wie zum Beispiel das Aufschließen des Autos, Anzeigen des Smartphone Displays oder Abspielen der Musik. Zur Kommunikation zwischen diesen fahrerspezifischen Diensten und den fahrzeugspezifischen Diensten, gibt es jedoch noch sehr wenig Ansätze. Um Sicherheit zu gewährleisten und eine schnelle Straßenzulassung zu erhalten, konzentrieren sich die Projekte auf eine klare Trennung der Kommunikation und einen reinen Lesezugriff auf fahrzeugbezogene Dienste. Dies bietet einen Ansatzpunkt für zukünftige Forschungsprojekte. Vor allem auf dem Visio.M Projekt der TU München kann aufgebaut werden, da diese bereits einen ähnlichen, allerdings stark limitierten, Ansatz gewählt haben. Solche Ansätze könnten mit einer Dienstarchitektur zum Beispiel durch WSDL oder OSGi umgesetzt werden.

5.1 Konferenzen und Experten

Im Automobilbereich gibt es zahlreiche Konferenzen zu den verschiedenen Teilbereichen. Im Bereich Kommunikation zwischen Systemen gibt es drei große Forschungsbereiche: Kommunikation im Auto (in-car) [21], Kommunikationen zwischen Autos (car-to-car) [13, 5, 40], Kommunikation zwischen Auto und der Umwelt (car-to-x)[13, 8]. Leider ist der Bereich einer neuen Systemarchitektur gerade erst in der Entstehung und es gibt noch keine Konferenzen im Automobilbereich dazu. Allerdings beschäftigt sich unter anderem die IAV [14] mit diesem Themengebiet.
In Bezug auf Dienstarchitekturen gibt es allerdings viele Konferenzen der W3C, ACM und IEEE, die sich mit Web Services befassen [1, 9, 15]. Aus diesem Bereich kann man viele Ansätze auf die Dienstwelt des Autos übertragen.

5.2 Reflexion und Ausblick

Im Verlaufe des Semesters habe ich viel im Bereich der Softwareentwicklung für Automobile gelernt. Aber vor allen Dingen, habe ich mich zum ersten Mal sehr ausführlich mit der Recherche befasst. Das war eine sehr gute Erfahrung und ich hoffe, dass ich mit dem gesammelten wissen im Grundprojekt weiterkomme.
Das Thema Schnittstellen und Interaktionen zwischen fahrer- und fahrzeugbezogenen Diensten ist ein aufkommendes Forschungsthema. Im Grundprojekt werde ich versuchen die Recherche Ergebnisse anzuwenden und die verschiedenen Technologien auszuprobieren. Es ist sicherlich möglich auf anderen Projekten wie dem Visio.M aufzubauen, jedoch ist die Interaktion und Kommunikation der verschiedenen Dienste noch ein weitestgehend unerforschter Bereich.

Anhang

1 IT-Infrastruktur des Visio.M

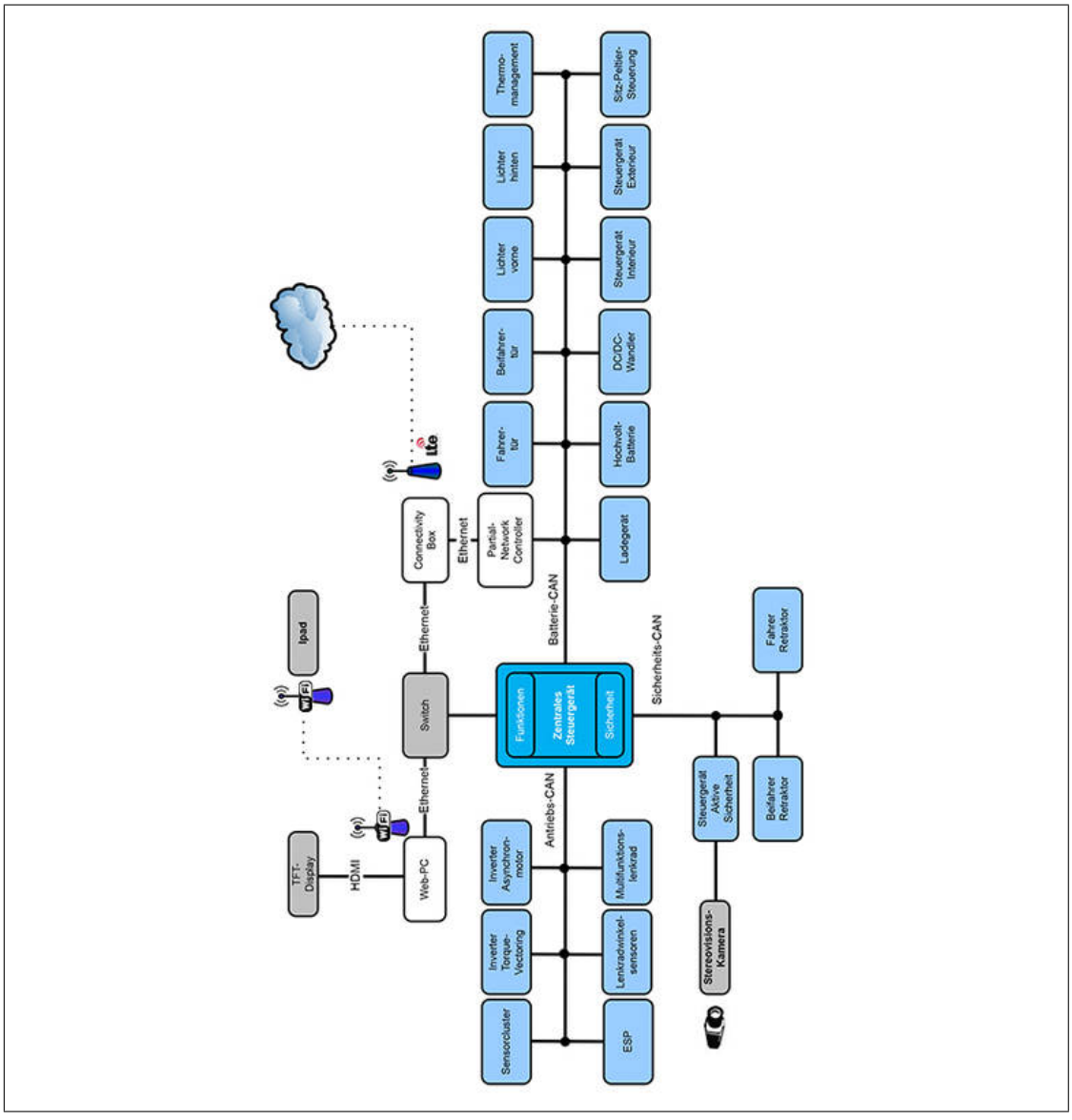

Abbildung 1: IT-Architektur des Visio.M Projektes der Technischen Universität München.[32]

Literaturverzeichnis

[1] : *ACM SAC 2017 - SOAP track*. – URL http://sac-soap.sdu.dk/soap2017/. – Zugriffsdatum: 2016-07-10

[2] : *Android Auto*. – URL https://www.android.com/intl/de_de/auto/. – Zugriffsdatum: 2016-07-03

[3] : *Automotive :: Hochschule Coburg*. – URL https://www.hs-coburg.de/forschung-kooperation/forschungsprojekte-oeffentlich/automotive.html. – Zugriffsdatum: 2016-07-14

[4] : *BMW ConnectedDrive : Übersicht*. – URL http://www.bmw.de/de/topics/faszination-bmw/connecteddrive/ubersicht.html. – Zugriffsdatum: 2016-07-03

[5] : *Car 2 Car - Communication Consortium: 2016*. – URL https://www.car-2-car.org/index.php?id=271. – Zugriffsdatum: 2016-07-10

[6] : *Car-Net. Die Apps und Dienste von Volkswagen.*. – URL http://volkswagen-carnet.com/de/de/start.html. – Zugriffsdatum: 2016-07-03

[7] : *Carplay*. – URL http://www.apple.com/ios/carplay/. – Zugriffsdatum: 2016-07-03

[8] : *COmmunication Network VEhicle Road Global Extension*. – URL http://converge-online.de/?id=010000&spid=de. – Zugriffsdatum: 2016-07-10

[9] : *ESOCC 2016*. – URL http://esocc2016.eu/. – Zugriffsdatum: 2016-07-10

[10] : *Extensible Markup Language (XML)*. – URL https://www.w3.org/XML/. – Zugriffsdatum: 2016-05-22

[11] : *Google Self-Driving Car Project*. – URL http://www.google.com/selfdrivingcar. – Zugriffsdatum: 2016-07-03

[12] : *Home - CoRE Group*. – URL http://core.informatik.haw-hamburg.de/. – Zugriffsdatum: 2016-05-19

[13] : *Homepage | 2016 IEEE Vehicular Networking Conference (VNC)*. – URL http://www.ieee-vnc.org/. – Zugriffsdatum: 2016-07-10

[14] : *IAV - Home*. – URL https://www.iav.com/. – Zugriffsdatum: 2016-06-30

[15] : *ICWS 2016*. – URL http://icws.org/2016/. – Zugriffsdatum: 2016-07-10

[16] : *Mercedes me connect*. – URL https://www.mercedes-benz.com/de/mercedes-me/konnektivitaet/. – Zugriffsdatum: 2016-07-03

[17] : *OSGi™ Alliance – The Dynamic Module System for Java.* – URL https://www.osgi.org/. – Zugriffsdatum: 2016-05-14

[18] : *OSGi Service Gateway Specification Release 1.0 - r1.osgi-spec.pdf.* – URL https://osgi.org/download/r1/r1.osgi-spec.pdf. – Zugriffsdatum: 2016-05-22

[19] : *Startseite - IAF - Institut für Automobil Forschung.* – URL http://www.automobil-forschung.org/de/. – Zugriffsdatum: 2016-07-14

[20] : *VDA.* – URL http://vda.de. – Zugriffsdatum: 2016-07-14

[21] : *VEHICULAR 2016.* – URL http://www.iaria.org/conferences2016/VEHICULAR16.html. – Zugriffsdatum: 2016-07-10

[22] : *Visio.M: Home.* – URL http://www.visiom-automobile.de/home/. – Zugriffsdatum: 2016-05-13

[23] : *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language - wsdl20-z.pdf.* – URL https://www.w3.org/TR/2006/CR-wsdl20-20060327/wsdl20-z.pdf. – Zugriffsdatum: 2016-05-22

[24] : *World Wide Web Consortium (W3C).* – URL https://www.w3.org/. – Zugriffsdatum: 2016-05-22

[25] 4.0, Kompendium I.: *Automobil - Industrie 4.0.* – URL http://plattform-maerkte.de/auto/. – Zugriffsdatum: 2016-05-19

[26] Barry, Douglas K. ; Dick, David: *Web services, service-oriented architectures, and cloud computing: the savvy manager's guide.* Second edition. San Francisco, Calif. : Oxford : Morgan Kaufmann ; Elsevier Science [distributor], 2013 (The Savvy manager's guides). – ISBN 978-0-12-398357-2

[27] Bean, James: *SOA and Web services interface design: principles, techniques, and standards.* Amsterdam ; Boston : Morgan Kaufmann/Elsevier, 2010 (Morgan Kaufmann OMG Press). – ISBN 978-0-12-374891-1

[28] Bosch, GmbH: *CAN Specification.* 1991. – URL http://www.bosch-semiconductors.de/media/ubk_semiconductors/pdf_1/canliteratur/can2spec.pdf. – Zugriffsdatum: 2016-07-14

[29] Broy, M.: Automotive software and systems engineering. In: *Proceedings. Second ACM and IEEE International Conference on Formal Methods and Models for Co-Design, 2005. MEMOCODE '05.*, Juli 2005, S. 143–149

[30] Broy, Manfred: Challenges in automotive software engineering. In: *Proceedings of the 28th international conference on Software engineering*, ACM, 2006, S. 33–42. – URL http://dl.acm.org/citation.cfm?id=1134292. – Zugriffsdatum: 2016-07-14

[31] Broy, Manfred ; Kruger, Ingolf H. ; Pretschner, Alexander ; Salzmann, Christian: Engineering automotive software. In: *PROCEEDINGS-IEEE* 95 (2007), Nr. 2, S. 356. – URL https://mediatum.ub.tum.de/doc/1251761/1251761.pdf. – Zugriffsdatum: 2016-07-14

[32] Dr. Battenberg, Andreas: *Das Auto als Internet-Hardware.* – URL https://www.tum.de/die-tum/aktuelles/pressemitteilungen/kurz/article/32277/. – Zugriffsdatum: 2016-05-14

[33] Dr. Weinmann, Ulrich: *Software im Automobil – Anforderungen und Chancen.* – URL http://www.seuh.org/SEUH8_2003/01_Weinmann.pdf. – Zugriffsdatum: 2016-05-23

[34] Hall, Richard ; Pauls, Karl ; McCulloch, Stuart ; Savage, David: *Osgi in Action: Creating Modular Applications in Java.* 1st. Greenwich, CT, USA : Manning Publications Co., 2011. – ISBN 978-1-933988-91-7

[35] He, W. ; Yan, G. ; Xu, L. D.: Developing Vehicular Data Cloud Services in the IoT Environment. In: *IEEE Transactions on Industrial Informatics* 10 (2014), Mai, Nr. 2, S. 1587–1595. – ISSN 1551-3203

[36] Iwai, A. ; Aoyama, M.: Automotive Cloud Service Systems Based on Service-Oriented Architecture and Its Evaluation. In: *2011 IEEE International Conference on Cloud Computing (CLOUD)*, Juli 2011, S. 638–645

[37] Papazoglou, M. P.: Service-oriented computing: concepts, characteristics and directions. In: *Proceedings of the Fourth International Conference on Web Information Systems Engineering, 2003. WISE 2003*, Dezember 2003, S. 3–12

[38] Pretschner, Alexander ; Broy, Manfred ; Kruger, Ingolf H. ; Stauner, Thomas: Software Engineering for Automotive Systems: A Roadmap. In: *2007 Future of Software Engineering*. Washington, DC, USA : IEEE Computer Society, 2007 (FOSE '07), S. 55–71. – URL http://dx.doi.org/10.1109/FOSE.2007.22. – Zugriffsdatum: 2016-05-19. – ISBN 978-0-7695-2829-8

[39] Ran, Li ; Junfeng, Wu ; Haiying, Wang ; Gechen, Li: Design method of CAN BUS network communication structure for electric vehicle. In: *2010 International Forum on Strategic Technology (IFOST)*, Oktober 2010, S. 326–329

[40] Sfez, Aurélie: *ETSI - Car 2 Car Communication Consortium.* – URL http://www.etsi.org/news-events/events/883-car-2-car-communication-consortium. – Zugriffsdatum: 2016-07-10

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 14. Juli 2016 Timo Häckel