



Informationstechnik

Hochschule Esslingen
University of Applied Sciences

IT-Innovationen

Band 16
Januar 2016

Grußwort des Dekans

Liebe Leserinnen und Leser,

Industrie 4.0 – Arbeit 4.0 – Gesundheit 4.0 sind nur einige der Schlagwörter die für die Digitalisierung der Gesellschaft und damit für die große Herausforderung der Zukunft stehen. Immer mehr nimmt sie Einzug in neue Lebensbereiche, immer stärker bestimmt sie unser Leben.



Ermöglicht und befeuert wird die Digitalisierung durch die enormen Fortschritte in der Informationstechnik und insbesondere durch die weltweite Vernetzung - das Internet. Jedem Gerät in einer Fabrik, in einem Kraftwerk, einer Bahnanlage, einem Krankenhaus, bis hin in den Privathaushalt zur Waschmaschine oder dem Sonnenrollo, also jedem noch so kleinen Ding, verschafft die Digitalisierung einen Zugang zum Internet, dem Internet of Things. Ganz neue Herausforderungen werden an die Informationstechnik herangetragen, wovon einigen eine besondere Bedeutung zukommt: dem Schutz der Geräte vor unberechtigtem Zugriff, sowie dem Datenschutz. Es bedarf gut ausgebildeter, hoch motivierter und pfiffiger junger Menschen, um die sich mit der Digitalisierung bietenden Möglichkeiten zu erkunden und verantwortungsvoll in unseren Alltag der Zukunft gewinnbringend für uns alle zu integrieren. Wir in der Fakultät Informationstechnik nehmen diese Aufgabe ernst und bilden junge Menschen unter dieser Prämisse aus. All ihr Wissen und Können fließt in die Abschlussarbeit, die Sie hier in diesen Ihnen vorliegenden IT-Innovationen sehen, von unseren Absolventen selbst dargestellt und zusammengefasst.

Viel Freude beim Lesen wünscht Ihnen herzlichst Ihr

A handwritten signature in blue ink that reads "Nonnast". The signature is written in a cursive, flowing style.

Prof. Jürgen Nonnast

Dekan der Fakultät Informationstechnik

IMPRESSUM

ERSCHEINUNGSORT

73732 Esslingen am Neckar

HERAUSGEBER

Prof. Jürgen Nonnast
Dekan der Fakultät Informationstechnik
der Hochschule Esslingen - University of Applied Sciences

REDAKTIONSANSCHRIFT

Hochschule Esslingen - University of Applied Sciences
Fakultät Informationstechnik
Flandernstraße 101
73732 Esslingen am Neckar

Telefon +49(0)711.397-4211
Telefax +49(0)711.397-4214
E-Mail it@hs-esslingen.de
Website www.hs-esslingen.de/it

REDAKTION, LAYOUT UND DESIGN

Prof. Dr.-Ing. Reinhard Schmidt
Hochschule Esslingen - University of Applied Sciences
Fakultät Informationstechnik
Flandernstraße 101
73732 Esslingen am Neckar

SATZ, ANZEIGEN und VERLAG

Fabian Müller, B. Eng.
Hochschule Esslingen - University of Applied Sciences
Fakultät Informationstechnik
Flandernstraße 101
73732 Esslingen am Neckar

ERSCHEINUNGSWEISE

Einmal pro Semester, jeweils Januar und Juni

DRUCK

Pixelgurus
Werbung – Werbetechnik – Digitaldruck.
Horbstraße 8
73760 Ostfildern

AUFLAGE

500 Exemplare

ISSN 1869-6457

Timo Acquistapace	Entwicklung einer Vorgehensweise für Studentenprojekte auf Basis der dualen Softwareentwicklung nach Bertrand Meyer	1
Alp Arslan Aslan	Konzeption einer Anfahrhilfe für Diesel- und Benzinfahrzeuge	3
Ilyas Atishev	Integration eines Bedien- und Anzeigesystems in die E/E-Architektur eines Elektrofahrzeugs	4
Philipp Barchet	Entwicklung einer Umgebung für 3D-Simulationen zur Automatisierung von Tests eines autonomen Reinigungsgerätes mittels der Strategie Software-in-the-Loop unter Verwendung des Simulationswerkzeugs V-REP einschließlich Erstellung einer API-Bibliothek in C++ und Visualisierung unter ROS	6
Büsra Bayrak	Prozessoptimierung basierend auf hardwareunabhängigen Laufzeitmessungen in der AUTOSAR Komponentenentwicklung	8
Vladimir Belyaev	Lokalisierung eines Fahrzeugs anhand von Odometriedaten basierend auf einem aus Kartendaten abgeleiteten gerichteten Graphen mit Hilfe eines Kalman Filters in Matlab	10
Simon Birk	Konzeption und Implementierung eines Analyse-Werkzeugs zur Identifikation von Code Smells auf Basis der .NET Compiler Plattform Roslyn	12
Tobias Brandt	Entwicklung eines Embedded-Profilers und einer automatisierten Benchmark für ein Echtzeitbetriebssystem nach AUTOSAR	14
Edgar Burghardt	Evaluation und Erweiterung von Echtzeitbetriebssystemen	16
Ali Celebi	Konzeption und Realisierung eines Software-Moduls für den Test der Head-Unit eines Automobils mittels Restbussimulation auf Basis eines bewährten Test-Frameworks	18
Tobias Claus	Integration eines Mikrocontroller-Motortreibers in den bürstenlosen Gleichstrommotor des Valeo Laserscanners „Scala 1403“	20
Marco D'Ottavio	Design und Entwurf eines automatisierten Testsystems für Multiprotokoll-Sensoren im Kontext von Continuous Integration mit Definition eines applikationsspezifischen XML-Schemas	22
Fayez Daas	Untersuchungen zur Entwicklung einer Diagnosestrategie für Smart NOx-Sensoren im Fahrzeugbetrieb	24
Tobias Dahmen	Konzeption und Design einer Webanwendung zur BAföG-Berechnung für Antragsteller und Sachbearbeiter	26
Siegfried Derksen	Konzipierung und Implementierung einer auf dem Client-Server-Modell basierenden Kommunikationsschicht zur Erweiterung einer generischen Plattform für die Entwicklung von Automatisierungskomponenten mit dem Ziel zeitgleicher logischer Verbindungen mehrerer Applikationen zu einem Feldgerät	28
Dharyl Christ Djigoue Pouani	Konzeption und Realisierung einer Anbindung von Mikrocontroller-basierter Hardware an mobiles Multimediagerät zur Datenvisualisierung	30
Philipp Dürnay	Fehlertolerante Systeme durch effektive Laufzeitdiagnose mit SFL	32
Christian Eissler	Konzeption und Realisierung der Erweiterung einer Seriensoftware um eine Schnittstelle zu CAN FD für ein als Gateway konzipiertes Automotive-Steuergerät einschließlich der Entwicklung des Hardware Abstraction Layers	35

Daniel Fröhlich	Entwurf und prototypische Realisierung der Software unter FreeRTOS zu einem Test-Interface für das AC-System von Festo	37
Jan Hammer	Vergleich der Protokolle AKEP2 und ECDH für fahrzeuginternen Peer-to-Peer Schlüsselaustausch	39
Tobias Michael Heine	Ermittlung der Fahrzeuglage mittels Sensordatenfusion	41
Daniel Herbst	Entwicklung von Komfort- und Servicefunktionen für ein „Neighborhood Electric Vehicle“	43
Christian Holder	Konzeption und Implementierung eines grafischen Editors zur Skriptverwaltung einer bestehenden Softwarelösung	45
Tobias Horn	Konzepterstellung zur sicheren Datenkommunikationsschnittstelle im Bereich Connected Cars	47
Steffen Huzel	Integration des Kommunikationsprotokolls DDS (Data Distribution Service) in einen flexiblen Protokollstapel für die Datenübertragung in Fahrzeugnetzwerken	49
Neşe Inan	Modellbildung zur echtzeitfähigen und beobachterbasierten Temperaturüberwachung für den Zwischenkreiskondensator eines Pulswechselrichters für Elektro- und Hybridfahrzeuge	51
Lukas Jaeckle	Das Architekturmuster Model-View-ViewModel	53
Markus Just	Stabile und evolvierbare Software durch Einhaltung der SOLID-Prinzipien	55
Daniel Kahrizi	Konzeption und prototypischer Aufbau eines mobilen Prüfsystems zur automatisierten Funktionsanalyse von Automotive-Aktuatorik-Komponenten	57
Marc Kaiser	Konzeption, Implementierung und Validierung einer Firewall in einer AUTOSAR Umgebung	59
Daniel Klaiber	Entwurf und prototypische Implementierung einer Umgebung zur massiven parallelen Simulation unter der Verwendung von GPGPU-Hardware	61
Simon Kuhn	Erstellung einer KI für ein Grand Strategy Spiel – Anwendung und Analyse verschiedener Entwicklungsprinzipien	63
Tanja Last	Entwicklung und Implementierung einer Anwendung für Microsoft SharePoint 2013 zur Beantragung und Genehmigung von Benutzerrechten für Microsoft Team Foundation Server unter Verwendung des Architekturmusters MVC im agilen Entwicklungsumfeld von Scrum	65
Christoph Lenger	Entwicklung eines XCP-Slaves zum Messen und Parametrieren eines Universalsteuergerätes	67
Wojciech Lesnianski	Sauberes Fahren – Teilsystem Simulation	69
Robert Lotter	Analyse und Vergleich verteilter Datenbanksysteme und Entwurf einer verteilten Online-Shop Datenbank.	71
Patrick Lutz	Entwicklung und Validierung eines adaptiv-markierenden Lichts zur selektiven Beleuchtung potentieller Gefahrensituationen	73
Patrick Mack	Konzipierung und prototypische Realisierung einer über magnetische Referenzpunkte geführten Navigation für ein fahrerloses Transportsystem	75

Daniel Majorow	Konzeption und Realisierung eines Fahrzeugrechners als Teil eines verteilten Systems zur Erfassung und Auswertung der Emissionswerte von Fahrzeugen	77
Thorsten Mollenkopf	Evaluierung verschiedener Datenbanken im Embedded bereich	79
Nikolas Porzer	Konzipierung und Realisierung eines Übertragungsmechanismus zum Austausch von Steuerbefehlen und Daten zwischen Geräten für den kathodischen Korrosionsschutz und einer zentralen Leitebene basierend auf UMTS und unter Verwendung der Übertragungsprotokolle SNMP und HTTP	81
Martin Pröbstle	Prototypische Realisierung einer closed-loop Testumgebung für eine Dosing-Control-Unit (SCR System) auf Basis µLabCar	83
Tankred Raitz	Integration von Smart Home Geräten in das homee System	85
Julian Rapp	Modellbasierte Längs- und Querregelung für ein autonom fahrendes Modellauto	87
Matthias Roth	Evaluation von OpenCL für FPGAs	89
Lukas Ruckwied	Use Case 2.0 – Ivar Jacobson 2011	91
Tobias Schmid	System on a Chip für stereobasierte Kamerasysteme Hardwareentwicklung und Programmierung der Kameratreiber in VHDL	93
Simon Schneider	Web Services in einer serviceorientierten Architektur	95
Natascha Schöneck	Fahrspuren- und Spurbreitenschätzung mit radarbasierten Fahrerassistenzsystemen	97
Jens Schuler	Entwurf und Realisierung der Diskretisierung des Simulink-Modells eines pneumatischen Linearantriebs von Festo sowie Integration des diskretisierten Modells in die Entwicklungsplattform	99
Peter Schulze	Analyse relevanter Kommunikationssysteme der Automatisierungstechnik hinsichtlich der Synchronisation mit dem unterlagerten AC-System von Festo sowie Verifikation der Ergebnisse für PROFINET, EtherCAT und Sercos III	101
Yunike Setiowati	Entwicklung eines Computerspiels mit Android	103
Vyacheslav Sobin	Konzeption und Realisierung eines MMI in einem verteilten System zur Erfassung und Auswertung der Emissionswerte von Fahrzeugen unter Einsatz von JavaFX und RESTful Web Services	105
Patrick Solisch	Analyse von Frameworks zur interaktiven Visualisierung großer Datenmengen im Web und Implementierung mit der JavaScript Bibliothek D3	107
Tino Julian Stadelmaier	Konzeption und Realisierung einer Datenbank und eines Web Servers zur Kommunikation mittels RESTful Web Services in einem verteilten System zur Erfassung und Auswertung der Emissionswerte von Fahrzeugen	109
Manuel Strehle	Hardware- und Software-Entwicklung der Steuerung einer Ladesäule für Elektrofahrzeuge nach DIN EN 61851-1	111
Anja Unnasch	Konzeption und Entwicklung eines Telematik Steuergeräts auf einem Heterogenem Multiprozessorsystem	113

Malte Viehweger	Entwicklung von AUTOSAR-Diagnosesoftwarekomponenten für ein Demonstrationsfahrzeug	115
Markus-Jonathan Wendler	Konsolidierung von AUTOSAR Steuergeräten in einer Linux Umgebung	117
Carmen Wettemann	Aufbau eines Simulationsframeworks in MATLAB für die Evaluation von Trackingalgorithmen für Parklinienmarkierungen	119
Marc Wöllper	Design and Implementation of an Encrypted Inter-Processor Communication for Embedded ARM Environments Using UML Based State Machine Code Generation	121
Christian Wolf	Kontinuierliche Selbstkalibrierung von Stereokameras	123

Entwicklung einer Vorgehensweise für Studentenprojekte auf Basis der dualen Softwareentwicklung nach Bertrand Meyer

Timo Acquistapace*, Joachim Goll, Manfred Dausmann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Agile Ansätze halten immer mehr Einzug in die Welt der Softwareentwicklung und sind in vielen Firmen und Projekten bereits zum Standard geworden. Der Ablauf agiler Prozesse unterscheidet sich erheblich von den zuvor eingesetzten spezifikationsorientierten Vorgehensweisen. Dieser Unterschied tritt vor allem im Hinblick auf eine durchdachte Planung, insbesondere Up-Front-Entscheidungen, und die Einbindung des Kunden zutage.

Studenten, welche sich während ihres Studiums immer wieder mit Gruppenprojekten befassen, müssen sich entscheiden, ob sie einem agilen oder einem spezifikationsorientierten Ansatz zur Umsetzung ihrer Projekte folgen sollen. Beide Herangehensweisen bieten sowohl Vor- als auch Nachteile, welche in der Folge diskutiert werden sollen. Im Zuge der durchgeführten Gruppen-Bachelorarbeit entstand ein kombinierter Ansatz, der nach Bertrand Meyer als „Duale Entwicklung“ zu charakterisieren ist und sich auch über Studentenprojekte hinaus hervorragend eignet.

Vorteile der beiden Vorgehensweisen

Agile Prozesse zeichnen sich dadurch aus, dass infolge einer iterativen, inkrementellen Entwicklung möglichst früh eine lauffähige Version des Systems bereitgestellt wird und somit auch in einer frühen Phase des Entwicklungsverlaufs Feedback vom Kunden zum System eingeholt werden kann. Der Kunde, bei Studentenprojekten in Gestalt des betreuenden Professors, kann somit frühzeitig Einfluss auf die Entwicklung nehmen, die Entwicklung verfolgen und hat durch das zyklisch ausgelieferte, lauffähige Produktinkrement einen ständigen Fortschrittsnachweis des Projekts. Die Architektur des Systems wird bei einer agilen Vorgehensweise nach und nach durch Refactoring gewonnen und optimiert.

Bei einem spezifikationsorientierten Vorgehen werden zunächst die Anforderungen aufgestellt und der Problembereich analysiert. Die daraus abgeleitete Architektur des Systems wird in einer Dokumentation festgehal-

ten. Sind die Anforderungen, die Systemanalyse und die Architektur vom Team und dem Kunden akzeptiert und wurde somit ein sogenanntes „Shared Understanding“ geschaffen, so startet der Implementierungsprozess. Dieser steht aufgrund der ausgiebigen Vorüberlegungen auf einer soliden Basis und kann dadurch meist sehr geradlinig und ohne große Änderungen an der Systemarchitektur verlaufen.

Nachteile der beiden Vorgehensweisen

Wird rein spezifikationsorientiert vorgegangen, so werden viele Up-Front-Entscheidungen getroffen und sehr viel Zeit in die Dokumentation investiert. Die Erfahrung hat gezeigt, dass Studenten zu viel spezifizieren und dokumentieren und dass dadurch letztendlich die Zeit für die Umsetzung der erdachten Features fehlt. Das eigentliche Software-Produkt wird erst zu einem späten Zeitpunkt ausgeliefert, sodass es auch keine Möglichkeit zu einem frühen Feedback gibt.

Durch die große Zeitspanne, welche für die der Implementierung vorausgehenden Planung benötigt wird, entsteht ein weiteres Problem: Die Spezifikation veraltet. Anforderungen können sich in der Zwischenzeit geändert haben und dadurch kann das modellierte System bereits obsolet sein. Zudem besteht die Möglichkeit, dass die Kommunikation zwischen dem Kunden und dem Requirements-Ingenieur durch Missverständnisse und Fehlinterpretationen negativ beeinträchtigt wurde und somit falsche Anforderungen festgehalten und modelliert wurden. Derartige Fehler werden bei einer spezifikationsorientierten Vorgehensweise zu spät bemerkt, wodurch die entstehenden Fehlerkosten ein beträchtliches Ausmaß annehmen.

Agile Prozesse hingegen sind flexibel gegenüber Änderungswünschen des Kunden bzw. gegenüber Änderungen der Anforderungen, solange diese zwischen den einzelnen Entwicklungszyklen eingebracht werden. Der Verzicht auf eine erste Planung und gene-

*Diese Arbeit wurde durchgeführt bei der Firma Firma IT-Designers GmbH, Esslingen-Zell

rell auf Up-Front-Entscheidungen birgt jedoch große Risiken, wenn ein komplexes System entwickelt werden soll. Es ist naiv, stets davon auszugehen, dass sich bei einer Erweiterung des ersten lauffähigen Inkrements die Komplexitäten lediglich addieren. Kommt es zu einer Zusammenarbeit der hinzugefügten Programmteile mit den bisher bereits implementierten, so kommt es meist auch zu einer sogenannten multiplikativen Komplexität, welche ohne vorherige Analyse des Gesamtsystems nur schwer handhabbar ist.

Generell bedarf ein Agiles Vorgehen großer Erfahrung bzw. einer kompetenten Anleitung durch erfahrene Softwareentwickler, die agile Verfahren sehr gut beherrschen und mit dem zu realisierenden Systemtyp bereits vertraut sind. Studenten-Projekte, welche agil durchgeführt werden, scheitern daher oftmals.

Die Duale Softwareentwicklung als erfolgreiche Kombination

Um die Vorteile beider Ansätze nutzen zu können, deren Nachteile jedoch auszumerzen, empfiehlt Bertrand Meyer die Duale Softwareentwicklung. In der frühen Phase eines Prozesses liegt hierbei der Fokus auf der Planung und auf Design-Entscheidungen, welche ein erweiterbares und skalierbares System mit der erforderlichen Infrastruktur garantieren. Diese Entscheidungen werden analog zu einer spezifikationsorientierten Vorgehensweise auf Basis initial aufgestellter Requirements und einer ersten Systemanalyse getroffen. Auch bei einer gewissen Unschärfe der initialen Anforderungen sollte es nach Meyer erfahrenen Softwareentwicklern möglich sein, derartige Entscheidungen zu fällen und das Projekt somit auf ein solides Fundament zu stellen.

Im weiteren Verlauf wird der Fokus auf die Entwicklung der Software gelegt und diese vorangetrieben, wobei regelmäßig Produktinkremente entwickelt und ausgeliefert werden. Das Refactoring der grundlegenden Architektur fällt, falls zuvor gute Vorüberlegungen und Design-Entscheidungen getroffen wurden, in seinem Umfang deutlich kleiner aus als bei einer rein agilen Vorgehensweise und dient nun eher als zusätzliches Hilfsmittel, denn als Allheilmittel.

Der hier angedeutete sequentielle Ablauf dieser beiden Schritte kann auch in einen parallelen Ablauf umgewandelt werden. Dadurch ist es möglich, dem Kunden erste Prototypen vorzustellen und dessen Feedback zu gewinnen, während parallel an der grundlegenden Systemarchitektur gearbeitet wird. Die Beteiligten beider Prozesse befinden sich in regel-

mäßigem Austausch, sodass deren Erkenntnisse kombiniert werden können. Dadurch wird sowohl erreicht, dass die erste Systemarchitektur den Kundenwünschen gerecht wird, als auch sichergestellt, dass Teile der ersten Prototypen auch später im Projekt verwendet werden können [1].

Einsatz der Dualen Softwareentwicklung in Studenten-Projekten

Die Idee der Dualen Softwareentwicklung in ihrer sequentiellen Variante eignet sich hervorragend für Studenten-Projekte. In der frühen Phase werden die Vision und die Ziele des Systems definiert sowie eine erste Architektur erstellt. In der zweiten Phase werden zyklisch mehrere Sprints durchlaufen, an deren Anfang stets die Aufstellung der Requirements und die Systemanalyse für die im Sprint zu entwickelnde Funktionalität steht. Daraufhin werden diese Features von den Gruppenmitgliedern umgesetzt und am Ende des Sprints ein Review durchgeführt, um die Ergebnisse, aber auch das Vorgehen selbst, zu bewerten und Verbesserungspotential aufzuzeigen.

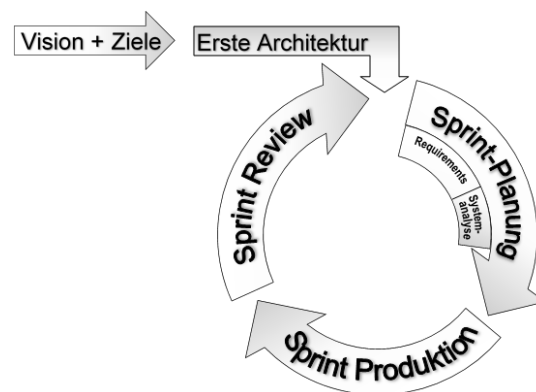


Abbildung 1: Gewähltes Vorgehensmodell

Dieses Vorgehen mag zunächst etwas an Scrum erinnern, da bei Scrum oft von einem sogenannten „Sprint 0“ die Rede ist, in welchem ebenfalls die Ziele des Systems und eine erste Architektur definiert werden. Dieser vorbereitende Sprint findet jedoch im offiziellen Scrum-Guide keine Erwähnung und auch bekannte Vertreter der Scrum-Philosophie wie Mike Cohn zweifeln die Sinnhaftigkeit dieses initialen Sprints an. Eine erste Architektur könne auch im ersten produktiven Sprint entstehen. Werden wichtige Entscheidungen schon vorab benötigt, so solle ein „Projekt vor dem Projekt“ eingeführt werden [2].

[1] Meyer, B. (2014). Dual Development. In B. Meyer, Agile! (S. 74–75). Zürich: Springer International Publishing.
 [2] Cohn, M. (05.03.13). Using Scrum on an Analysis Project: <https://www.mountain-goatsoftware.com/blog/using-scrum-on-an-analysis-project>, Zugriff am: 10.11.15

Konzeption einer Anfahrhilfe für Diesel- und Benzinfahrzeuge

Alp Arslan Aslan*, Reiner Marchthaler, Hermann Kull

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die heutige Mobilität führt zu hohen Kohlenstoffdioxid-Emission und Ressourcenverbrauch, unabhängig von der Art des Antriebs. Neue Techniken für effizientere Kraftstoffverbrennung helfen, die Ressourcen zu schonen und den Kohlenstoffdioxid-Ausstoß zu senken. Einer dieser technischen Fortschritte ist das Downsizing-Konzept. Damit sollen sowohl die hohen Erwartungen des Kunden an Leistung sowie die von dem Gesetzgeber immer weiter verschärften Emissionsrichtlinien erfüllt werden. Downsizing steht für die Reduzierung des Motorhubvolumens mit gleichzeitiger Steigerung der Leistung beziehungsweise des Drehmoments bei kleinerem Hubraum. Maßgeblich für eine Steigung der Leistung ist die Anhebung der Nenndrehzahl oder die Erhöhung des maximalen effektiven Mitteldrucks mittels Abgasturboaufladung und der direkten Kraftstoffeinspritzung. Dabei wird eine größere Menge Luft-Kraftstoffgemisch bei gleichem Zylindervolumen effizienter verbrannt. Dadurch genügt ein kleinerer Hubraum, um die gleiche Leistung freizusetzen als bei einem konventionellen Motor. Downsizing-Konzepte haben den Vorteil einer geringeren Reibleistung und einer geringeren Motorgewichts. All dies führt zur signifikanten Reduktion des Kraftstoffverbrauches und damit des Kohlenstoffdioxid-Ausstoßes. [1], [2]

Neben den Vorteilen bringt das Downsizing-Konzept neue Herausforderungen mit sich. Das Anfahrverhalten wird beim turboaufgeladenem Motor, aufgrund der geringen Ansaugluft im Leerlaufbereich schlechter. Das Anfahren beschreibt die Überbrückung der Drehzahlücke aus dem Stillstand und gehört zu den Kernaufgaben eines Getriebes. In Fahrzeugen mit automatisch schaltenden Getrieben kann der Fahrer seinen Beschleunigungswunsch nur über die Betätigung des Fahrpedals äußern (siehe Abbildung 1). In Fahrzeugen mit manuell geschalteten Getrieben wird der Beschleunigungswunsch durch aufeinander abgestimmten Betätigungen des Fahr- und Kupplungspedals geäußert. Dabei kann der gekoppelte Ablauf dieser beiden Pedale nicht optimal verlaufen. Kommen die gegenläufigen Effekte des Downsizing-Konzeptes hinzu, führt dies zu einem ruckartigen Anfahren, ggf. zum Abwürgen des Motors. Diese Bachelorarbeit befasst sich mit der Erstellung eines Konzepts für eine Anfahrunterstützung durch die Leerlaufregelung bei Diesel- und Benzinfahrzeugen. Hierzu werden zuerst die Einflussgrößen für den Anfahrvorgang ermittelt sowie deren Wirkung auf das Fahrverhalten. Darauf aufbauend wird das Konzept erstellt und dieses Konzept für ein Dieselfahrzeug validiert.

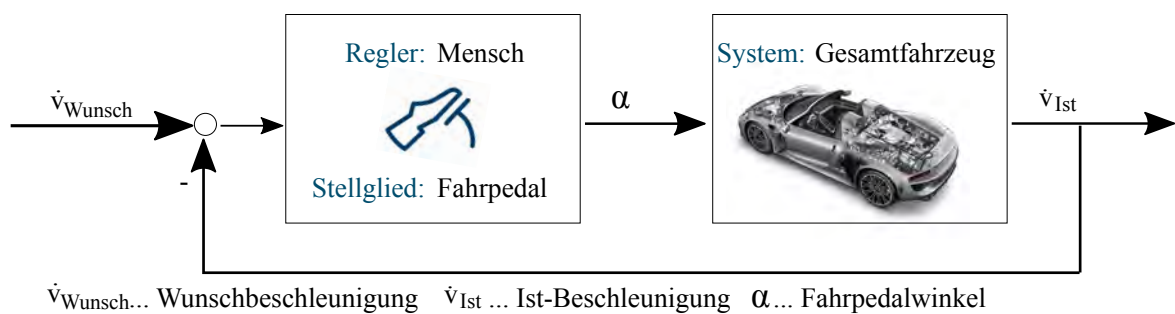


Abbildung 1: Fahrereingabe zur Realisierung einer Wunschbeschleunigung

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Schwieberdingen

[1] Konrad Reif, Kraftfahrtechnisches Taschenbuch, Vieweg+Teubner Verlag, 27. Auflage, ISBN 3834814407
[2] Prof. Dr.-Ing. Hermann Kull, Manuskript der Vorlesungreihe KFZ-Steuergeräte-Design, WS2013

Bildquellen:

- Abbildung 1: Eigene Darstellung

Integration eines Bedien- und Anzeigesystems in die E/E-Architektur eines Elektrofahrzeugs

Ilyas Atishev*, Werner Zimmermann, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In den letzten Jahren sieht man eine steigende Tendenz zum Einsatz von digitalen Kombiinstrumenten. Anstatt der konventionellen analogen Instrumente werden frei konfigurierbare Displays eingesetzt. Sie bieten mehr Flexibilität als herkömmliche Systeme und können in Kombination mit der Mittelkonsole im Fahrzeug zur Bedienung von Fahrerassistenzsystemen verwendet werden.



Abbildung 1: Beispiel eines Kombiinstrumentes

In dieser Arbeit wird ein Bedien- und Anzeigesystem in Form eines digitalen Kombiinstrumentes für ein Neighborhood Electric Vehicle (NEV) entwickelt. Der Begriff NEV entstand in den USA in den 90-er Jahren des vergangenen Jahrhunderts [1]. NEV ist ein Leichtelektrofahrzeug, das üblicherweise auf Basis eines Golfcarts aufgebaut ist. Ein NEV eignet sich gut für Kurzfahrten in Wohngebieten. Durch den Elektromotor entstehen weniger Schallemissionen als bei einem Auto mit einem Verbrennungsmotor. Dank der einfachen Bedienung eignet es sich besonders gut für ältere Menschen.

Die GIGATRONIK Stuttgart GmbH entwickelt seit einigen Jahren ihr eigenes NEV. Die Basis dafür ist das Golfcart EZGO Freedom RXV. Das NEV ist mit diversen Fahrerassistenzsystemen ausgestattet, die von Studenten entwickelt werden. Es dient als Forschungsplattform für angehende Ingenieure und Techniker. Im Laufe der Zeit wurde das Golfcart umgebaut und verfügt nun über diverse Fahrerassistenzsysteme und Erweiterungen. Diese sind:



Abbildung 2: Das NEV der GIGATRONIK Stuttgart GmbH

Bereits integrierte Systeme:

1. 360 Grad Umgebungserkennung
2. Automatischer Einparkassistent
3. Automatischer Notbremseassistent
4. Rückfahrkamera
5. Steer-By-Wire-Lenkung mit Lenkrad oder Sidestick
6. Force-Feedback am Lenkrad
7. WLAN-Anbindung zur Fernsteuerung
8. LED-Scheinwerfer mit Abblendlicht, Kurvenlicht, Blinklicht und RGB-LED-Tagfahrlicht.
9. 48V Bordnetz mit erweiterten Spannungsebenen um 12V und 24V
10. Geschwindigkeitsregelanlage

*Diese Arbeit wurde durchgeführt bei der Firma GIGATRONIK Stuttgart GmbH, Stuttgart

Aktuell in der Entwicklung:

1. Kamerabasierte Fahrbahn- und Fahrspurerkennung
2. Fußgängererkennung mittels Infrarotsensorik
3. Cloudbasierte Fahrzeugdiagnose

Aufgrund der Vielfalt der Teilsysteme entstand im Fahrzeug eine komplexe E/E-Architektur mit mehreren Bussystemen. Eine der Herausforderungen dieser Arbeit besteht in der Integration des aufzubauenden Systems in die E/E-Architektur des Fahrzeugs. Des Weiteren wird eine Entscheidung getroffen, welches Betriebssystem und welche Technologie für die Softwareentwicklung einzusetzen sind. Im nächsten Schritt werden die Bussysteme des Fahrzeugs untersucht und am Arbeitsplatz simuliert. Parallel dazu wird eine Anwendung mit graphischer Benutzeroberfläche entwickelt. Im weiteren Schritt werden Anforderungen an ein modernes Kombiinstrument festgelegt. Es wird eine graphische Benutzeroberfläche unter Berücksichtigung der heutigen Standards erstellt. Zum Schluss wird das aufzubauende System in das Fahrzeug integriert. Es wird geprüft, ob das System in verschiedenen Fahrszenarien und Fällen funktionsfähig ist. Um die Ziele der Arbeit genauer zu definieren, wird eine Liste von funktionalen Anforderungen erstellt. Im Folgenden werden die funktionalen Anforderungen stichwortartig vorgestellt:

- **Anzeigen des Kombiinstrumentes**

1. Geschwindigkeit
2. Antriebsleistung
3. Batterieladezustand
4. Visualisierung der Rekuperation
5. Gangwahl

6. Farbwahl des Tagfahrlichts
7. Kilometerstand
8. Fahrtrichtungsanzeiger
9. Datum und Uhrzeit

- **Bedienung der Fahrerassistenzsysteme**

1. Parksensoren
2. Rückfahrkamera
3. Geschwindigkeitsregelanlage
4. Abschleppmodus
5. Darstellung der Fahrspurerkennung

- **Fahrzeugdiagnose**

1. Anzeige von Spannungen, Strömen und Temperaturen
2. CAN-Aufzeichnung

Die für das Kombiinstrument relevanten Botschaften werden über zwei CAN-Netzwerke übertragen. Für die Kommunikation mit dem CAN-Bus wird eine Anwendung in der Programmiersprache C++ entwickelt. Bei der Entwicklung wird inkrementell vorgegangen. Es werden gleichzeitig mehrere Teilsysteme entwickelt. Nach der Einführung von jeder wichtigen Funktion wird das System in Betrieb genommen und die Funktionalität wird verifiziert. Während der Implementierung der geforderten Funktionen wird die graphische Benutzeroberfläche entworfen und programmiert. Im Ergebnis wird ein digitales Kombiinstrument entwickelt, das in der Lage ist, mit mehreren Steuergeräten im Fahrzeug zu kommunizieren und wichtige Fahrzeugdaten anzuzeigen. Über den Touchscreen werden mit diesem Kombiinstrument auch Fahrerassistenzsysteme bedient.

[1] United States Patent USD378994: Veröffentlicht am 17. Juli 1995.

Bildquellen:

- Abbildung 1: <http://www.audiusa.com/models/audi-tt-coupe>
- Abbildung 2: Intranet der GIGATRONIK Stuttgart GmbH

Entwicklung einer Umgebung für 3D-Simulationen zur Automatisierung von Tests eines autonomen Reinigungsgerätes mittels der Strategie Software-in-the-Loop unter Verwendung des Simulationswerkzeugs V-REP einschließlich Erstellung einer API-Bibliothek in C++ und Visualisierung unter ROS

Philipp Barchet*, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die mobile Service- und Reinigungsrobotik ist seit längerem nicht mehr nur an Hochschulen und Universitäten ein Gesprächsthema. Im Handel sind verschiedenste Endgeräte für die private und industrielle Nutzung verfügbar. Diese Roboter müssen sich längst nicht mehr nur in vorpräparierten Umgebungen zurecht finden, sondern müssen auf neue Umgebungen oder zufällige, plötzliche Ereignisse

dynamisch reagieren können. Um sich in unbekannten Umgebungen zurecht zu finden, benötigen diese Roboter unterschiedlichste Sensorik. Die Entwicklung von Sensorsystemen und auch der Algorithmen zur Verarbeitung der Daten und Generierung der Befehle für die Bewegungen ist sehr schwierig und zeitaufwändig.

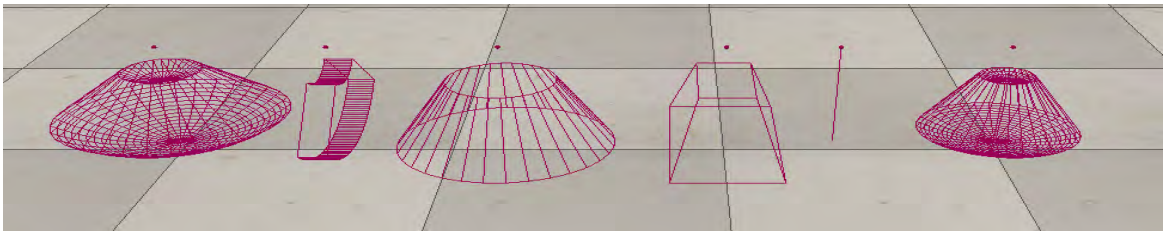


Abbildung 1: Keulen von Ultraschallsensoren

Gerade das Testen von einzelnen Modulen oder Änderungen an Algorithmen ist bei der Entwicklung eines der wichtigsten Punkte, um das Gesamtsystem robust und funktionsfähig zu machen. Der Aufbau von Simulationsumgebungen ist in diesem Fall das meist gewählte Mittel. Zum Vorantreiben der Entwicklung wird die Strategie Software-in-the-Loop verwendet. Im Fokus der Simulation steht somit die möglichst genaue Wiedergabe von Sensor- und Kinematikeigenschaften. Ein Simulationswerkzeug, welches es ermöglicht, ein Modell und eine entsprechende Simulationsumgebung aufzubauen, ist das Programm *Virtual Robot Experimentation Platform*, kurz V-REP. Dieses bietet bereits verschiedene Sensormodelle wie etwa Keulen von Ultraschallsensoren, welche in Abbildung 1 zu sehen sind. Alle abgebildeten Keulen ermöglichen eine Konfiguration ihrer Sichtwinkel und der minimalen und maximalen Reichweite und werden von V-REP bereit gestellt. Auch andere Sensorarten wie beispielsweise Lasersensoren

lassen sich so simulieren. Zudem ist die visuelle Anzeige der Laserstrahlen möglich, wie in Abbildung 2 dargestellt ist.

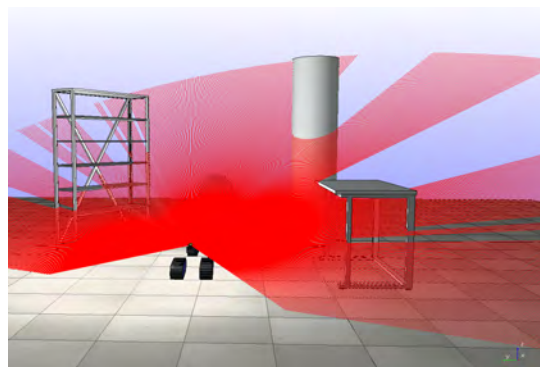


Abbildung 2: Lasersensorebene

Das Testen neuer Einstellungen kann in den meisten Fällen schneller in einer Simulation durchgeführt werden. Neue Sensorik und Aktuatorik ist mit wenigen Handgriffen in das

*Diese Arbeit wurde durchgeführt bei der Firma Alfred Kärcher GmbH & Co. KG, Winnenden

Modell aufgenommen, während echte Bauteile oft mühsam an dem realen Prototypen befestigt werden müssen. Durch sogenannte *embedded Skripte* und viele APIs (Application Programming Interface) kann die Simulation

und das Roboter-Modell gesteuert und ohne viel Aufwand in bestehende Projekte integriert werden. Eine Übersicht der möglichen Schnittstellen zeigt Abbildung 3.

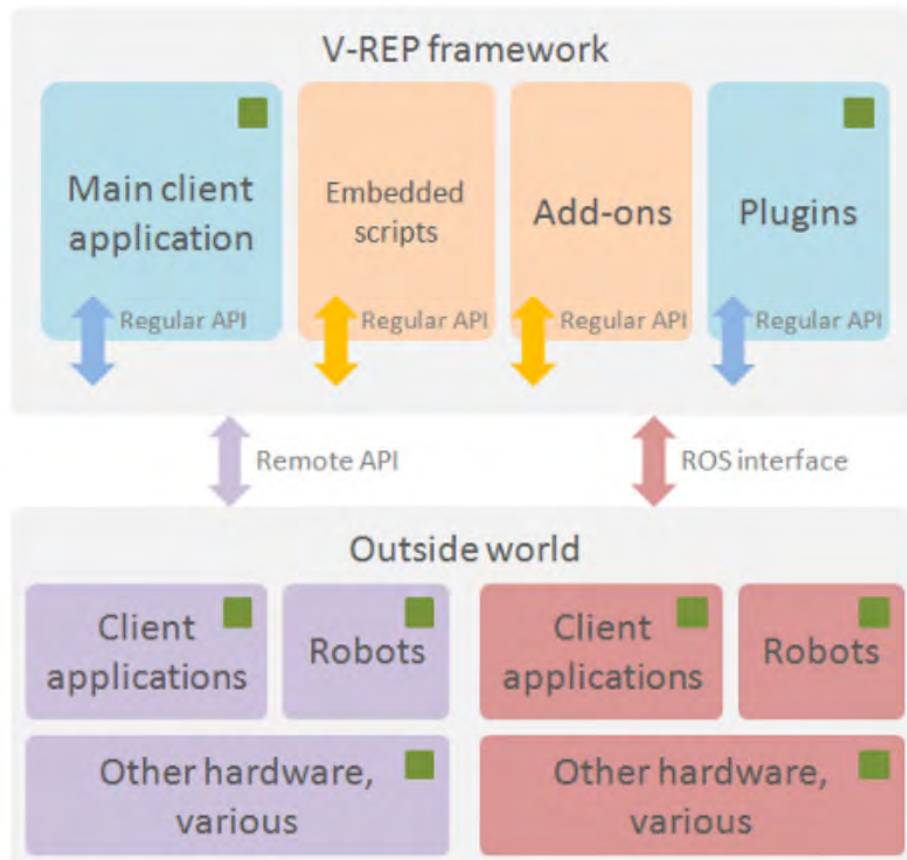


Abbildung 3: Schnittstellen V-REP

Im Laufe dieser Arbeit wurde eine komplexe Simulationsumgebung mit mehreren Bereichen aufgebaut, welche sich im Detaillierungsgrad der Objekte innerhalb der Simulation unterscheiden und es ermöglichen, die Algorithmik nach und nach immer realeren Situationen anzupassen. Zudem wurde auch ein Robotermodell erarbeitet, welches einen existierenden Prototypen als Vorlage hatte. Um sowohl mit der Simulation zu kommunizieren und Einstellungen wie Hindernisse einzufügen oder zu

entfernen und auch mit dem Robotermodell zu interagieren, wurde eine *remote API-Bibliothek* in C++ ausgearbeitet. Die Bibliothek ist in einzelne Funktionsbereiche unterteilt, die alle Aspekte der Roboterinteraktion abdeckt. So ist die Steuerung über einen externen Motorcontroller möglich und zudem unabhängig von der Visualisierung der Sensordaten. Beides ist über die *remote API-Bibliothek* möglich.

[1] Coppelia Robotics GmbH: User Manual www.coppeliarobotics.com/helpFiles/ Stand 10.2015

Bildquellen:

- Abbildung 1: Philipp Barchet
- Abbildung 2: Philipp Barchet
- Abbildung 3: Coppelia Robotics GmbH [1]

Prozessoptimierung basierend auf hardwareunabhängigen Laufzeitmessungen in der AUTOSAR Komponentenentwicklung

Büdra Bayrak*, Manfred Dausmann, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Im Laufe der Zeit wurden in der Automobiltechnik viele neue Funktionen entwickelt, um einen höheren Sicherheitsstandard und Komfort zu gewährleisten. Ein Ende ist nicht in Sicht, im Gegenteil, die Forschung im Bereich Kommunikationstechnologie hat ebenfalls begonnen. Mit steigender Komplexität der Steuergeräte-Software, steigt ebenfalls der Prozessor-Ressourcenverbrauch. Da dieser stark begrenzt ist, wird bereits in der Komponentenentwicklung nach Lösungsansätzen gesucht, die eine optimale Nutzung der vorhandenen Ressourcen gewährleisten sollen und die Qualitätseigenschaften steigern. Laufzeitmessungen wurden aufgrund dessen zu einem wichtigen Bestandteil in der Entwicklungsphase von Steuergeräte-Software. Um diese durchzuführen wurden verschiedene Methoden und Tools entwickelt. [2]

Es kann periodisch ein Speicherauszug der CPU erstellt werden, um zu überprüfen, welche Funktion einer Software in diesem Moment abgearbeitet wird. Problem hierbei ist jedoch, dass Funktionen mit sehr kurzen Laufzeiten übersehen werden können, wenn diese zwischen zwei Perioden abgearbeitet werden.

Eine weitere Methode ist das Auslesen von sogenannten 'Clock Ticks'.

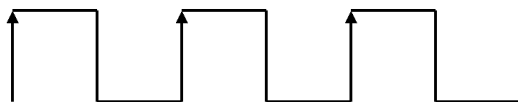


Abbildung 1: Clock Signal

Diese Clock Ticks sind in Abbildung 1 dargestellt. Sie werden verwendet, um Prozesse in einem Betriebssystem zu synchronisieren. Die Ticks treten periodisch auf und sind in den meisten Betriebssystemen auf 100 MHz konfiguriert. CPUs führen Instruktionen in diesen Clock Ticks aus. Ein System mit einer höheren Tick Rate arbeitet schneller, da die CPU Instruktionen dadurch schneller abgearbeitet werden können. In der Laufzeitmessung werden die verstrichenen Ticks aufsummiert und zum Schluss durch die Frequenz geteilt, um

eine Angabe in einer Zeiteinheit zu erhalten.[1][4]

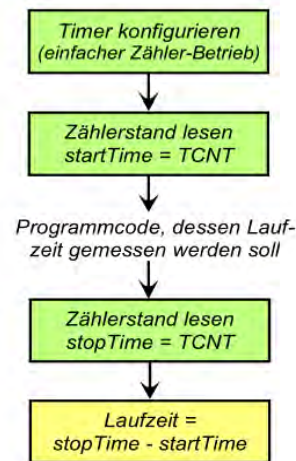


Abbildung 2: Ablauf einer Laufzeitmessung

Abbildung 2 zeigt den Verlauf der Laufzeitmessung. Zunächst muss der Zähler zurückgesetzt werden. Anschließend werden Start- und Endzeit definiert. Der Programmcode, welcher gemessen wird, befindet sich zwischen den gesetzten Grenzen. Zum Schluss werden die Ticks ausgelesen und die Differenz gebildet. Sollten jedoch während der Laufzeit Prozesse mit höherer Priorität zwischenzeitig auftreten, werden diese ebenfalls in die Laufzeitanalyse integriert und verfälschen somit das Ergebnis.

Um die Performance von Programmen und Prozessen zu steigern, werden kürzlich verwendete Instruktionen im Cache Speicher abgelegt. Diese werden bei erneutem Ausführen eines Programmes aus dem Zwischenspeicher geladen und sorgen somit für eine schnellere Ausführung des Programmes. Ein Cache Hit oder Cache Miss kann nicht beeinflusst werden und sorgt somit für unterschiedliche Messergebnisse in der Laufzeitmessung. Pipelining und Branch Prediction sorgen ebenfalls für kürzere Ausführungszeiten einer Applikation und somit zu verfälschten Messergebnissen. [1] [2] Neben diesen Messmethoden gibt es unterschiedliche Tools, welche Laufzeita-

*Diese Arbeit wurde durchgeführt bei der Firma Vector Informatik GmbH, Stuttgart

analysen bereits automatisiert durchführen und ein Instrumentieren des Programmcodes seitens des Programmierers nicht benötigt wird. Welche der unterschiedlichen Laufzeit Mess-techniken letztendlich eingesetzt wird, hängt stark von den Anforderungen an die Software ab und weshalb eine Laufzeitanalyse durchgeführt wird.

Eine Analyse dieser Art ist jedoch nicht immer ausreichend, besonders wenn das zu untersuchende System Sicherheitsanforderungen erfüllen muss. Hierbei handelt es sich um Echtzeitanforderungen. Ein Programm oder eine Funktion muss korrekte Ergebnisse in einer definierten Zeit liefern. Sollte diese Zeit überschritten werden, kann dies katastrophale Folgen haben. Daher wird eine Analyse der Worst Case Execution Time (WCET) einer Software benötigt.

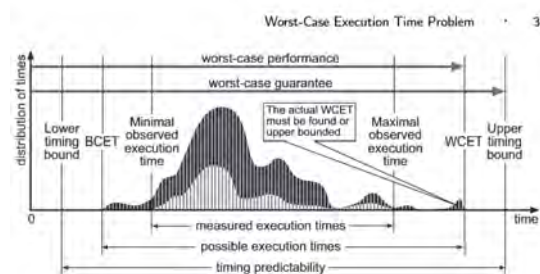


Abbildung 3: Worst Case Execution Time

Die WCET gibt Auskunft über die längste Laufzeit, die ein Programm annehmen kann. Dabei wird genau der Pfad untersucht, der den schlechtesten Fall einer Software darstellt und

in dem ein maximaler Zeitverlust erzielt wird.

In Abbildung 3 ist die WCET dargestellt. Die Best Case Execution Time (BCET) ist die geringste Laufzeit einer Software. Beide, BCET und WCET, definieren die Grenzen der maximal bzw. minimal möglichen Laufzeit, die ein System erzielen kann. Besonders wichtig ist, dass die WCET sich unterhalb einer definierten maximalen Grenze befindet. Bei dieser Grenze handelt es sich um eine harte oder um eine weiche Echtzeitbedingung. Die Verletzung einer weichen Grenze hat, bei seltenem Auftritt, nur geringe bis keine Folgen. Das System arbeitet jedoch nicht, wie von Echtzeitsystemen erwartet, zuverlässig. Die Verletzung einer harten Grenze kann jedoch fatale Folgen mit sich ziehen, bei denen Personenschäden entstehen können.

Um die WCET zu ermitteln, wurden verschiedene Verfahren entwickelt, doch keine ist in der Lage eine Zeit explizit anzugeben. Dafür muss nämlich der schlechteste Fall, den eine Software ablaufen kann, bekannt sein. Je komplexer das System ist, umso schwerer ist es dies herauszufinden. [3]

In dieser Arbeit werden die unterschiedlichen Laufzeit-Messtechniken evaluiert und auf einer AUTOSAR Basissoftware Komponente getestet. Dabei sollen verschiedene Konfigurationen und unterschiedliche Releases miteinander verglichen und eine Statistik über die Entwicklung dieser Komponente erstellt werden. Eine Analyse der WCET soll ebenfalls durchgeführt werden.

-
- [1] Harald Richter, TU Clausthal, Elektronik und Datenkommunikation im Automobil, 2005, <https://www.in.tu-clausthal.de/fileadmin/homes/techreports/ifi0905richter.pdf>, Zugriff: 20.10.15
- [2] Nat Hillary, Freescale Semiconductor, Measuring Performance for Real-Time Systems, http://www.freescale.com/files/soft_dev_tools/doc/white_paper/CWPERFORMWP.pdf, Zugriff: 30.10.15
- [3] Reinhard Wilhelm, Jakob Engblom, Andread Ermedahl, The Worst-case Execution Time Problem – Overview of Methods and Survey of Tools, 2012, <http://moss.csc.ncsu.edu/~mueller/ftp/pub/mueller/papers/1257.pdf>, Zugriff: 03.11.15
- [4] Gabriel Torres, Clock, 2005, <http://www.hardwaresecrets.com/how-a-cpu-works/2/>, Zugriff: 20.10.15

Bildquellen:

- Abbildung 1: <http://www.hardwaresecrets.com/how-a-cpu-works/2/>
- Abbildung 2: Prof. Dr. J. Friedrich, Prof. Dr.-Ing. Werner Zimmermann: Peripheriekomponenten, Ein-Ausgabeprogrammierung, Interrupts, VZ4.5 Stand August 2013
- Abbildung 3: <http://moss.csc.ncsu.edu/~mueller/ftp/pub/mueller/papers/1257.pdf>

Lokalisierung eines Fahrzeugs anhand von Odometriedaten basierend auf einem aus Kartendaten abgeleiteten gerichteten Graphen mit Hilfe eines Kalman Filters in Matlab

Vladimir Belyaev*, Walter Lindermeir

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

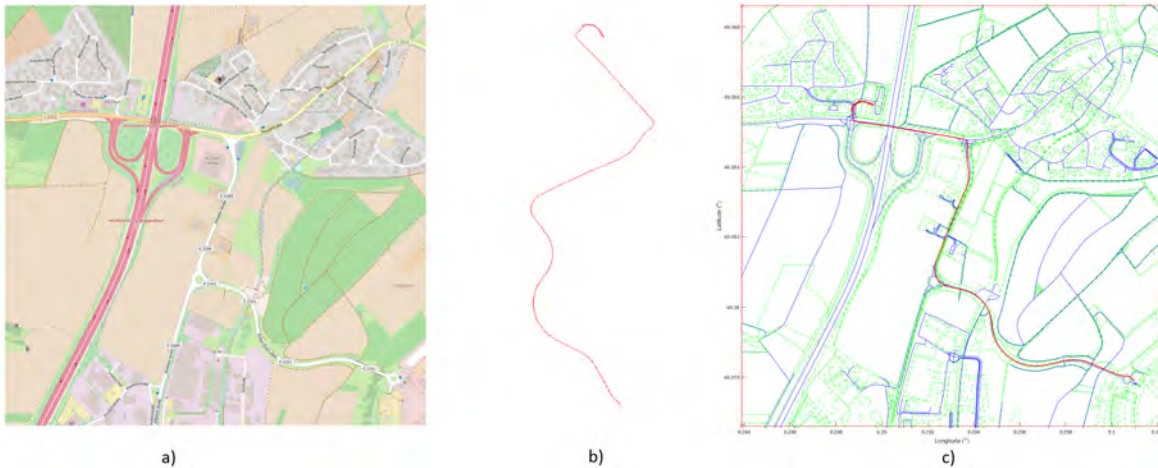


Abbildung 1: Darstellung der Fahrzeugbewegung auf der Karte

Die Entwicklung von Fahrerassistenzsystemen wurde bereits in den 70er Jahren mit der Erfindung des Antiblockiersystems (ABS) angefangen. Heutzutage kann der Abstand und die Geschwindigkeit des Fahrzeugs an das vorausfahrende Auto automatisch angepasst werden, was beim langen Stop-and-go-Fahren im Stau sehr hilfreich sein kann [1]. Es werden immer neuere Systeme entwickelt, die dem Fahrer mehr Fahrkomfort und Fahrsicherheit gewährleisten können. Die Entwicklung geht aber weiter und in der Zukunft erwarten uns autonome fahrende Fahrzeuge. Die technischen Entwicklungen in dieser Branche machen das autonome Fahren in der nahe Zukunft möglich. Bereits heute gibt es Fahrzeugprototypen, die autonom auf der Autobahn fahren können und die selbständig Entscheidungen bezüglich Geschwindigkeit und Fahrspurwechsel treffen [2].

Beim autonomen Fahren ist es erforderlich, die Fahrzeugposition zu jedem Zeitpunkt hochpräzise zu kennen. Ohne exakte Lokalisierung wäre es zum Beispiel nicht möglich, den Weg zum gewünschten Ziel sowie diverse Fahrmanöver zu planen. Außerdem kann die Position viele andere nützliche Informationen wie zum Beispiel den Weg bis zum nächsten Ausfahrt, zur nächsten Kreuzung oder bis

zur nächsten Wendemöglichkeit liefern. Dazu gehören auch solche Informationen wie Geschwindigkeitsbegrenzung, Anzahl der Spuren und mögliche Gefahrstellen. Alle diese Informationen sind notwendig für ein sicheres und komfortables Fahren.

Die Bestimmung der Position eines Fahrzeugs kann auf verschiedene Arten erfolgen. Normalerweise wird dafür das GPS (Globales Positionsbestimmungssystem) benutzt, das aber genauso wie alle anderen Systeme und Sensoren keine hundertprozentige Genauigkeit garantieren kann. Das GPS-Signal kann beim Fahren in einem Tunnel oder in einem stark bebauten Gebiet mit vielen Hochhäusern fehlen oder stark beeinträchtigt werden. Außerdem kann das GPS-Signal laut vielen Studien leicht verfälscht werden [3]. Als Folge kann das Fahrzeug von der Strecke gebracht werden oder an ein falsches Ziel navigiert werden.

Aus diesem Grund werden für die Positionserkennung redundante Quellen gebraucht. Das Ziel dieser Arbeit ist die Möglichkeit der Verwendung von Odometrie bei der Lokalisierung eines Fahrzeugs zu untersuchen. Das Wort „Odometrie“ kommt aus dem Griechischen und bedeutet „Wegmessung“. In Rahmen

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Abstatt

dieser Arbeit werden die Lenk- und Beschleunigungssensoren in einem autonom fahrenden Fahrzeug für die Schätzung der Position und der Orientierung in einem dreidimensionalen Raum verwendet. Die Daten aus diesen Sensoren werden von einem Prozessor ausgewertet und in kartesische Koordinaten konvertiert. Aus diesen Koordinaten kann anschließend der von einem Fahrzeug zurückgelegte Weg kalkuliert werden. In Abbildung 1b ist ein Beispiel für einen zurückgelegten Weg zu sehen. Es ist zu beachten, dass die Odometriedaten nur die relative Änderung der Fahrzeugposition bezüglich einer Startposition darstellen. Wenn die Startposition eines Fahrzeugs zum Beispiel mittels GPS ermittelt wird, wird es auch möglich, die Fahrzeugbewegung auf der Karte zu verfolgen. In Abbildung 1c ist eine in Matlab generierte Karte dargestellt. Der dazugehörige Kartenausschnitt aus OpenStreetMap ist in Abbildung 1a angezeigt. Die blauen Linien in Abbildung 1c stellen Landstraßen und Autobahnen dar. Mit grünen Linien sind kleine Straßen und Gebäudemarkierungen gekennzeichnet. Die rote Linie beschreibt den Fahrverlauf eines Fahrzeugs ausgehend von einer bekannten Startposition wie es aus gemessenen Odometriedaten einer Testfahrt ermittelt

wurde. Es ist ersichtlich, dass die Odometriedaten für eine Positionsbestimmung gut geeignet sind. Wie es aber oben erwähnt wurde, sind die GPS-Koordinaten auch bezüglich der Startposition nicht zuverlässig, da diese verfälscht werden können. Demzufolge kann nicht davon ausgegangen werden, dass die Fahrzeugposition am Anfang der Fahrt bekannt ist. Alle Möglichkeiten der Fahrzeugposition sollen zuerst gleichmäßig auf der ganzen verfügbaren Karte verteilt werden. Im Laufe der Fahrt werden alle Straßen ausgeschlossen, die zu dem bereits zurückgelegten Weg nicht passen. Zum Beispiel, wenn es aus den Odometriedaten bekannt ist, dass das Fahrzeug 400 Meter geradeaus gefahren und danach zwei Mal nach links abgebogen ist, werden alle Straßen ausgeschlossen, auf denen dieser Wegverlauf nicht möglich ist. Je länger das Fahrzeug fährt, desto weniger Straßen bleiben, die zu dem zurückgelegten Weg passen. Auf lange Sicht wird es nur eine Straße geben, die genau dem aus den Odometriedaten kalkulierten Weg entspricht. Somit können die Odometriedaten als unabhängige und redundante Quelle zur Lokalisierung dienen.

-
- [1] Bosch Mobility Solutions, „Bosch Media Service,“ 09.01.2013. [Online]. Verfügbar: http://www.bosch-presse.de/presseforum/details.htm?txtID=6071&tk_id=108&locale=de [Zugriff am 26.11.2015].
- [2] Bosch Mobility Solutions, „Bosch Media Service,“ 03.05.2013. [Online]. Verfügbar: http://www.bosch-presse.de/presseforum/details.htm?txtID=6235&tk_id=10 [Zugriff am 26.11.2015].
- [3] L. C. Christian Just, „ComputerBild,“ 01.08.2013. [Online]. Verfügbar: <http://www.computerbild.de/artikel/cb-Aktuell-Sicherheit-Wissenschaftler-leiten-80-Mio-Yacht-mit-gefaelschtem-GPS-Signal-in-die-Irre-8581122.html> [Zugriff am 26.11.2015].

Bildquellen:

- Abbildung 1a: OpenStreetMap. Verfügbar: <http://www.openstreetmap.org/#map=16/49.0828/9.2903> [Zugriff am 26.11.2015].
- Abbildung 1b,c: Eigene Darstellung in Matlab

Implementierung eines Analyse-Werkzeugs zur Identifikation von Code Smells auf Basis der .NET-Compiler-Plattform Roslyn

Simon Birk*, Andreas Rößler, Reinhard Schmidt

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

„Clean Code“ von Robert C. Martin ist derzeit eines der meist diskutierten und einflussreichsten Bücher über Code-Qualität. Darin werden anhand von Fallstudien Best Practices, Heuristiken und Code Smells identifiziert. Der Autor möchte ein Bewusstsein für die Problematiken schaffen, die durch schlecht geschriebenen Code entstehen: unsaubere Programme können mit der Weiterentwicklung zunehmend verwahrlosen und werden so immer anfälliger für Fehler. Die Implementierung neuer Features nimmt dadurch immer mehr Zeit in Anspruch. Ein Code Smell ist ein oberflächliches Symptom einer meist tiefer liegenden Problematik des Systems. In der Regel sind Code Smells leicht zu erkennen, wie zum Beispiel eine Methode, die aus vielen Zeilen Quelltext besteht. Allerdings müssen Code Smells nicht immer ein Indikator für Probleme sein. Manchmal ist es durchaus in Ordnung eine lange Methode zu schreiben – es kommt auf den Kontext an [1].

Das Ziel dieser Arbeit ist, ein Programmpaket von Code-Analysewerkzeugen zu entwickeln, mit deren Hilfe einige von Martin definierte Code Smells identifiziert werden können. Dies geschieht auf Basis der .NET-Compiler-Plattform Roslyn: Diese bietet umfassende Möglichkeiten, um Code zu analysieren und zu generieren.

Mit der Veröffentlichung von Roslyn schlägt Microsoft neue Wege ein: Die meisten Compiler sind als Black Box-Systeme implementiert – Entwickler haben keinen Zugriff auf den Funktionsumfang der einzelnen Compiler-Module. Roslyn bietet Schnittstellen, mit denen Quelltext auf syntaktische und semantische Aspekte untersucht werden kann. Folgende Grafik zeigt den schematischen Aufbau der in vier Komponenten aufgeteilten Compiler-Pipeline, sowie den dazugehörigen APIs:



Abbildung 1: Systemarchitektur von Roslyn

Zunächst zerlegt der **Parser** den Quelltext in die Bestandteile der Programmiersprache. Diese werden in einem sogenannten Syntax-Baum abgelegt, der eine vollständige Repräsentation des Codes darstellt. Anschließend werden alle benannten Symbole in einer Symboltabelle abgelegt. Der **Binder** verknüpft nun die im Syntax-Baum gespeicherten Bezeichner mit den Einträgen der Symboltabelle. Das Ergebnis ist ein semantisches Modell des Codes. Schließlich werden aus allen gewonnenen Informationen durch den **Emitter** die fertigen Assemblies generiert [2].

Mit Hilfe der Roslyn-API wird das Analysewerkzeug dieser Arbeit entwickelt. Das Tool ist in der Lage, einige der gängigsten Smells aus Martins Buch zu identifizieren, darunter auch die folgenden:

Ringabhängigkeiten in Vererbungshierarchien: Beim Entwurf objektorientierter Systeme werden oft Vererbungshierarchien entworfen. Dabei ist es wichtig, dass die Basisklasse unabhängig von den abgeleiteten Klassen ist, damit das Konzept der Basisklasse von den Konzepten der abgeleiteten Klassen getrennt ist. Sind in einer Basisklasse Abhängigkeiten zu einer von ihr abgeleiteten Klasse vorhanden, so ist möglicherweise mit dem Code etwas nicht in Ordnung [3].

Flag-Argumente: Die Übergabe eines booleschen Arguments ist ein klarer Hinweis, dass in der betreffenden Funktion mehr als nur eine Aufgabe erfüllt wird. Mit dem Zustand des Arguments wird über eine Kontrollstruktur bestimmt, welcher Code-Pfad innerhalb der Funktion ausgeführt wird. In den meisten Fällen sollte auf Flag-Argumente verzichtet und stattdessen zwei Methoden geschrieben werden [3].

Auskommentierter Code: Oft wird Code in dem Glauben auskommentiert, er würde nochmals gebraucht werden. Meistens ist dies nicht der Fall. Der auskommentierte Code bleibt bestehen, bläht den restlichen Quelltext unnötig auf und erschwert die Lesbarkeit. Code sollte nicht auskommentiert, sondern gelöscht werden! Sollte der Code nochmals gebraucht werden, kann eine ältere Version aus dem Versionskontrollsystem bezogen werden [3].

*Diese Arbeit wurde durchgeführt bei der Firma IT-Designers GmbH, Esslingen

Verletzungen des Law Of Demeters: Das LoD besagt, dass ein Modul nichts über das Innere der Objekte wissen sollte, die es manipuliert. Das LoD ist erfüllt, wenn eine Methode m einer Klasse K nur Methoden der folgenden Komponenten aufruft:

- K ,
- ein Objekt, das von m erstellt wird,
- ein Objekt, das als Argument an m übergeben wird,
- ein Objekt, das eine Instanzvariable von K ist,

Bei Verstößen gegen das LoD weiß die aufrufende Funktion zu viel über die innere Struktur der Objekte, mit denen sie arbeitet. Dadurch werden vermehrt Abhängigkeiten geschaffen. Verletzungen des LoD werden auch als *Train Wreck* (Zugkatastrophe) bezeichnet, weil die Struktur der aneinander gereihten Aufrufe wie ein zusammengekoppelter Eisenbahnwagen aussieht: `string street = person.Address.Street.ToString(); [3]`

Abgesehen von regelbasierten Bewertungen von Code kann dieser mit Hilfe von Metriken auch quantitativ bewertet werden. Es handelt sich dabei um Kennzahlen, die eine Aussage über verschiedene Aspekte des Codes treffen. Mit Metriken kann Quellcode verschiedener Projekte oder Arbeitsstände formal verglichen werden.

Die Kennzahl **Lines Of Code (LOC)** gibt beispielsweise an, aus wie vielen Zeilen Code eine Softwareeinheit besteht. Dabei ist allerdings nicht definiert, wie Kommentare oder Leerzeilen behandelt werden. Neben dem einfachen LOC-Maß gibt es noch erweiterte Definitionen, die diese Fragen klären.

Number Of Methods (NOM) gibt an, aus wie vielen Methoden eine Klasse besteht und **Number Of Classes (NOC)**, aus wie vielen Klassen ein Projekt besteht. Die **zyklomatische Komplexität (CC)** gibt an, aus wie vielen linear unabhängigen Pfaden eine Softwareeinheit besteht. Die Metrik lässt sich wie folgt bestimmen: Die Anzahl möglicher Pfade durch eine Softwareeinheit (entspricht der Anzahl von Kontrollstrukturen und Schleifen) wird mit eins addiert.

Durch die Kombination von Metriken können neue Kennzahlen bestimmen werden, mit

denen sich auch qualitative Aussagen über den Code treffen lassen: Beispielsweise bestimmt der Quotient

$$\frac{NOM}{NOC}$$

die durchschnittliche Anzahl an Methoden je Klasse – ein Anzeichen für die Kohäsion von Klassen. Ein hoher Wert lässt vermuten, dass die betreffende Klasse zu viele Verantwortlichkeiten übernimmt.

Die durchschnittliche Anzahl von Codezeilen pro Methode

$$\frac{LOC}{NOM}$$

beschreibt den Umfang von Funktionen. Hohe Werte lassen auf zu große Funktionen schließen, denn „Funktionen sollten kaum jemals länger als 20 Zeilen sein“ [3]. **Die durchschnittliche zyklomatische Komplexität pro Codezeile**

$$\frac{CC}{LOC}$$

bestimmt die Dichte von Verzweigungspunkten im Code. Mit Hilfe dieser drei Quotienten lassen sich Aussagen über den Umfang von Klassen und Methoden sowie über die Komplexität eines Projekts treffen [4]. Um statistische Vergleichswerte für C#-Projekte zu haben, wird im Rahmen dieser Arbeit ein **Metrik-Rechner** erstellt. Mit diesem Tool werden 32 Open Source-Projekte mit insgesamt über 2,5 Millionen Zeilen Code untersucht. Für jedes der Projekte werden die oben genannten Quotienten errechnet, kumuliert und anschließend die Schwellenwerte ermittelt. Die Ergebnisse sind in der folgenden Tabelle zusammengetragen:

Metrik	Gering	Durchschn.	Hoch	Sehr Hoch
CC/LOC	0,16	0,30	0,44	0,66
LOC/Methode	3	8	13	20
NOM/Klasse	3	7	11	17

Abbildung 2: Schwellenwerte von C#-Projekten

Mit den ermittelten Daten lässt sich schnell ein Überblick über eine Softwareeinheit gewinnen und macht sie mit anderen vergleichbar. Werden diese Kennzahlen für das eigene Projekt berechnet, ist anhand der Schwellenwerte schnell erkennbar, ob und bei welchen Klassen und Methoden Refaktorisierungsbedarf besteht.

[1] <http://martinfowler.com/bliki/CodeSmell.html>

[2] <https://github.com/dotnet/roslyn/wiki/Roslyn>

[3] Robert C. Martin, Clean Code

[4] Radu Marinescu und Michele Lanza, Object-Oriented Metrics in Practice

Bildquellen:

- Abbildung 1: <https://github.com/dotnet/roslyn/wiki/Roslyn>

Entwicklung eines Embedded-Profilers und einer automatisierten Benchmark für ein Echtzeitbetriebssystem nach AUTOSAR

Tobias Brandt*, Reiner Marchthaler, Werner Zimmermann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In dieser Arbeit werden ein schlanker Profiler und Benchmarks für ein AUTOSAR Echtzeitbetriebssystem entwickelt. Der Profiler vermisst die Performance mit automatisierten Benchmarks alle Betriebssystemschnittstellen und Kontextwechselzeiten. Dabei werden die Laufzeiten der einzelnen Funktionen dokumentiert und anschließend in einem Report ausgegeben. Anhand der Ergebnisse erhalten die Entwickler direktes Feedback über die Laufzeit ihrer Implementierungen und können gegebenenfalls Optimierung durchführen.

Um den steigenden Anforderungen und der wachsenden Komplexität in den Steuergeräten Herr zu werden, wurde der AUTOSAR – Standard geschaffen. AUTOSAR, ist eine Entwicklungspartnerschaft führender OEMs und Tier-1-Lieferanten [1]. Der Hauptgedanke ist, dass die Anwendungssoftware (ASW) unabhängig von der Hardwarearchitektur und deren Basissoftware (BSW) sein muss (siehe Abbildung 1). Durch diese Trennung können ASW und BSW unabhängig voneinander entwickelt werden und die Portierbarkeit der ASW steigt. [2]

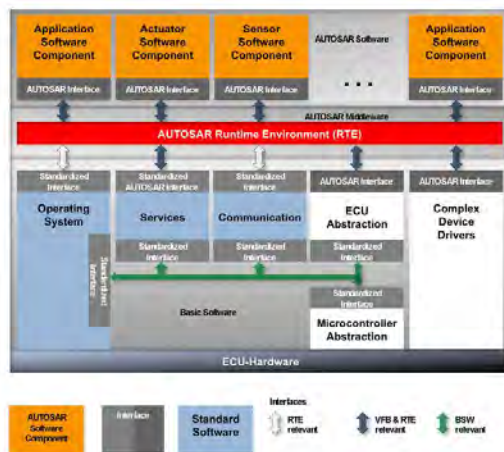


Abbildung 1: AUTOSAR Architektur

Eine zentrale Komponente in der BSW ist das Betriebssystem. Es bearbeitet sämtli-

che Kommunikationsprimitive, Kontextwechsel und zeitlich gesteuerten Aufgaben innerhalb der BSW. Betriebssystemprimitive werden von vielen AUTOSAR Komponenten intensiv genutzt. Aus diesem Grund geht der Laufzeitoverhead, der durch das Betriebssystem entsteht, signifikant in die Gesamtlaufzeit mit ein [3]. Um den Anforderungen eines Echtzeitbetriebssystems gerecht zu werden und eine schnelle Verarbeitung garantieren zu können, will man die Auslastung der CPU so gering wie möglich halten. Gerade bei eingebetteten Systemen ist der Ressourcenverbrauch (das Laufzeitverhalten und der Speicherverbrauch) eines Betriebssystems wettbewerbsentscheidend, da nur begrenzte Ressourcen zur Verfügung stehen.

Die Laufzeit auf der Hardware ist von weiteren Faktoren abhängig. Diese sind die Hardware, Konfigurationen des Betriebssystems, der Compiler und die verwendeten Compileroptionen. Bei der Hardware steht eine Vielzahl an unterschiedlichen Plattformen zur Verfügung mit unterschiedlichen Ressourcen (Speichergröße, Prozessorarchitektur, CPU Geschwindigkeit, Cachegröße, Anzahl der Register etc.). Weitere Faktoren wie z.B. der Cache, die Pipeline oder die Branch Prediction müssen ebenso berücksichtigt werden. Bei der Konfigurierbarkeit des Betriebssystems spielen die verschiedenen Schutzstufen (SC1-SC4) eine große Rolle, da dies zusätzlichen zu durchlaufenden Code bedeutet. Auch sind die Laufzeiten vom verwendeten Compiler und Compileroptionen abhängig. Wie stark der Compiler optimiert, ist hierbei entscheidend. Die Laufzeiten in den unterschiedlichen Varianten zu vermessen würde enormen Aufwand erfordern. Die Lösung muss dementsprechend automatisiert sein.

Das Vermessen soll anhand einer Profiler-Software umgesetzt werden. Ein Profiler ist ein Programmierwerkzeug, welches in der Softwareentwicklung Anwendung findet und das Laufzeitverhalten von Software analysiert. Er dient dem Softwareentwickler als Analysehilfsmittel um laufzeitintensive Abschnitte in einem

*Diese Arbeit wurde durchgeführt bei der Firma Vector Informatik GmbH, Weilimdorf

Programm aufzudecken.

Im ersten Teil dieser Arbeit wurden bereits existierende Profiler (Gprof, Valgrind) betrachtet und auf ihre Anwendbarkeit im AUTOSAR Umfeld untersucht. Jedoch wurden diese Tools für untauglich erklärt, da sie nur mit großen Einschränkungen auf Mikrocontrollern einsetzbar sind.

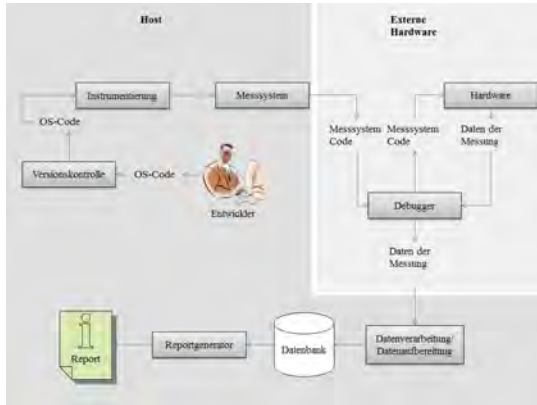


Abbildung 2: Profiler Konzept

Im zweiten Teil wurde dann eine eigene Lösung für einen schlanken und auf der Hardware einsetzbaren Profiler entwickelt. Die wichtigsten Anforderungen nach denen sich dieses Konzept in Abbildung 2 und somit auch der zu entwickelnde Profiler entwickelt haben waren: 1) Instrumentierung 2) Messungen 3) Weiterverarbeitung 4) Report.

Die Instrumentierung in dieser Arbeit erfolgt mit einem selbst geschriebenen Instrumentierungstool, das sich die AUTOSAR Makros zu Nutze macht. Nach der Instrumen-

tierung vermisst der Profiler die Betriebssystemschnittstellen und deren Unterfunktionen automatisch durch zuvor entwickelte Benchmarks-Tests, wobei das Messprogramm sämtliche Laufzeiten protokolliert. Die protokollierten Rohdaten werden so weiterverarbeitet, dass daraus direkt ein Bericht erstellt werden kann. Der Bericht soll Veränderungen sichtbar machen, um gegebenenfalls Optimierungen durchzuführen. Der Profiler wird automatisiert durch ein Continuous Integration Tool gestartet sobald sich Änderungen in der Versionskontrolle ergeben. Nach dem Start, beginnt der Profiler die einzelnen Schritte (siehe Abbildung 2) bis zum Report (siehe Abbildung 3) abzuarbeiten. Dieser Profiler bietet dem Entwickler die Möglichkeit sich voll und ganz auf die Implementierung zu konzentrieren und durch lediglich Einchecken des Codes eine schnelle Rückmeldung über die Laufzeit seiner Anwendung zu erhalten.

API	Tendenz	Aufrufhäufigkeit	Zeit (Ø) [µs]	Standardabweichung [µs]
Activate Task	▲	100/100	1.7	0.007
Terminate Task	▼	100/100	2.5	0.005
...

Funktion	Tendenz	Aufrufhäufigkeit	Zeit (Ø) [µs]	Standardabweichung [µs]
Activate Task				
Funktion 1	▲	40/100	0.3	0.0008
Funktion 1	▲	40/100	0.3	0.0008
Funktion 2	▼	20/100	0.7	0.0007

Abbildung 3: Beispiel Bericht

[1] AUTOSAR, www.autosar.org

[2] Werner Zimmermann – Bussysteme in der Fahrzeugtechnik: Protokolle, Standards und Softwarearchitektur

[3] Angelika Langer – Die Kosten der Synchronisation, www.angelikalanger.com

Bildquellen:

- Abbildung 1: www.autosar.org
- Abbildung 2: Eigene Darstellung
- Abbildung 3: Eigene Darstellung

Evaluation und Erweiterung von Echtzeitbetriebssystemen

Edgar Burghardt*, Peter Väterlein, Harald Melcher

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In eingebetteten Lösungen steigt der Anspruch an die Funktionalität permanent an. Dies führt zu zusätzlichen Ansprüchen an die Softwarekomponente des Systems. Insbesondere die Gewährleistung der Echtzeitfähigkeit, Portabilität, Wartbarkeit, Wiederverwertung und die Erweiterbarkeit der Software stellen eine Herausforderung für den Programmierer dar, wenn es um die Entwicklung und Umsetzung der Architektur geht.

Die Leistung des Systems alleine, kann keine definierte Antwortzeit garantieren. Auf einer ARM Architektur basierende Controller bieten eine Menge an Features, die bei Einsatz eines Echtzeitbetriebssystems ohne große Performanceverluste garantierte Antwortzeiten ermöglichen [1]. Die Auswahl sowie die Verfügbarkeit solcher Rechentechnik und die Zahl der Einsatzgebiete in eingebetteten Systemen werden zusehends größer.

Als eine Middleware, stellt ein Echtzeitbetriebssystem einen Kompromiss zwischen allen genannten Anforderungen dar. Durch die Bereitstellung von Schnittstellen und Mechanismen zur effizienten Ressourcenverwaltung, kann das Echtzeitbetriebssystem einen erheblichen Beitrag zur Minimierung der Reaktionszeit des Gesamtsystems beitragen. Das Echtzeitbetriebssystem stellt einen Scheduler zur priorisierten Verwaltung von Tasks bereit. Durch Multitasking entstehende Kontextwechsel entstehen zeitliche Verzögerungen zwischen den Ereignissen im System. Gleichzeitig wird jedoch gewährleistet, dass diese Unterbrechungen beim Scheduling angemessen berücksichtigt werden.

Ereignisse, wie externe Interrupts sowie systeminterne Vorgänge zur Wahrung der Datenintegrität, können diesen zeitlichen Verzug vergrößern. Dadurch wird ein Echtzeitbetriebssystem für den Einsatz in Echtzeitsystemen mit harten Echtzeitanforderungen oft untauglich.

Der Markt von Echtzeitbetriebssystemen wird von vielen kommerziellen Herstellern, als auch von Open Source Projekten bedient. Die angebotenen Echtzeitbetriebssysteme haben unterschiedliche Leistungsmerkmale. Die Angaben der Hersteller zu den Leistungsdaten der Systeme sind allerdings nicht ausreichend,

um eine qualifizierte Auswahl eines Betriebssystems zu ermöglichen.

Es existieren mehrere Methoden zur Evaluierung des durch ein Echtzeitbetriebssystem entstehenden Overheads wie RheaStone Benchmark [2], MiBench [3] oder Hartstone Benchmark [4]. Diese liefern jedoch nur ein Ergebnis in Form eines Durchschnitts. Solche gemittelten Ergebnisse sind wenig aussagekräftig und nicht verwertbar für die Planung konkreter Aufgaben [4]. Es wird ein Benchmark benötigt, der der aktuellen Anwendung angepasst ist.

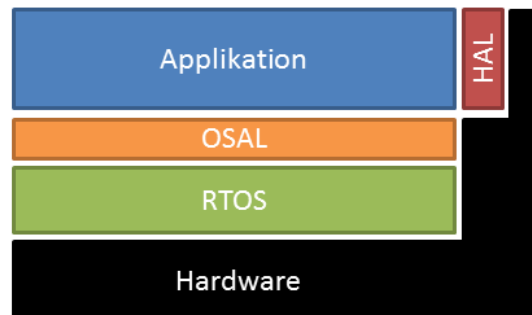


Abbildung 1: Aufbau der Applikation.

Deshalb soll eine Methode für die Evaluierung von Echtzeitbetriebssystemen entwickelt werden, um den Einfluss der externen Ereignisse auf die maximale Ausführungszeit des Gesamtsystems zu ermitteln. Ebenso soll es diese Methode erlauben, Anwendungen und Testfälle unabhängig vom Echtzeitbetriebssystem zu gestalten. Ein Open Source Echtzeitbetriebssystem soll mit dieser Methode analysiert werden und dessen Performance, in Bezug auf die Einhaltung harter Echtzeitbedingungen, verbessert werden. Es wird erwartet dass die Verwaltung der Prioritäten Hauptursache für den maximalen Verzug sind.

Die in dieser Arbeit verwendete Anwendung, dient der Steuerung eines Gelenks in einem Industriemanipulator. Dieses Gelenk enthält zwei Motoren. Versagt das Gelenk auf Grund der Überschreitung der zugelassenen Masse, so muss das System in der Lage sein, mit minimalem Verzug zu reagieren, um den entstehenden Schaden zu begrenzen. Ereignisse wie eine A/D Wandlung, Überschreitung

*Diese Arbeit wurde durchgeführt bei der Firma TZM GmbH, Göppingen

eines Stromwerts, oder Erreichen der Endposition des Gelenkes, zusammen mit dem ohnehin ständig mitlaufenden Systemtaktgeber des Prozessors, können bei gleichzeitigem Auftreten zu starken Verzögerung in der Ausführung der verschiedenen Tasks führen.

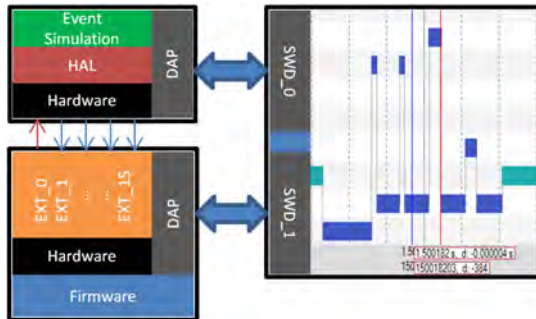


Abbildung 2: Aufbau des Messsystems

Das Evaluierungssystem besteht aus zwei Hardwarekomponenten. Die erste Komponente stellt das Zielsystem dar. Die Softwareapplikation des Zielsystems benutzt ein Echtzeitbetriebssystem als Middleware, das zur Verteilung von Ressourcen zwischen den einzelnen Aufgaben dient. Des Weiteren kann das Zielsystem über 16 Kanäle Unterbrechungen empfangen. Jede Unterbrechung leitet eine entsprechende Reaktion ein und kann unterschiedlich priorisiert werden.

Zur Gewährleistung der Unabhängigkeit der Softwareapplikation vom verwendeten Echtzeitbetriebssystem, wird eine einheitliche Betriebssystem Abstraktionsschicht (OSAL) entwickelt. Der OSAL bietet allgemeine Basisfunktionen eines Echtzeitbetriebssystems zur Erzeugung von Tasks, Taskmanagement, sowie Interprozesskommunikation. Der OSAL ermöglicht es, ohne Veränderungen der Applikation, zwischen Echtzeitbetriebssystemen zu wechseln. Abbildung 1 zeigt den Aufbau der Firmware des Zielsystems. Die Applikation hat keinen direkten Zugriff auf die Hardware. Der Hardware Abstraction Layer (HAL) abstrahiert

die Hardware von der Applikation und steigert die Portabilität. Der Operating System Abstraction Layer (OSAL) abstrahiert das Echtzeitbetriebssystem (RTOS) von der Applikation und erlaubt den schnellen Tausch der Echtzeitbetriebssysteme. Des Weiteren kann der OSAL zur Entwicklung neuer Applikationen verwendet werden. Die angebotenen Basisfunktionen, tragen zur Verbesserung der Portabilität der Anwendung bei.

Abbildung 2 veranschaulicht den Gesamtaufbau des Messsystems. Die zweite Komponente ist ein Stimulator, der in der Lage ist, gezielte, sowie zufällige Ereignisse für das Zielsystem zu erzeugen. Dank des Stimulators können unterschiedliche Szenarien für das Zielsystem simuliert werden. Durch unterschiedliche Stimulationen besteht die Möglichkeit, die Reaktionszeit des Zielsystems zu prüfen und zu beurteilen.

Zielsystem und Stimulator sind über 16 Externe Interrupt Kanäle des Zielsystems (EXT) miteinander gekoppelt und synchronisiert. Die Synchronisation erlaubt es, die Reaktion des Echtzeitbetriebssystems von der Erzeugung bis zum Erkennen der Unterbrechung und Einleitung einer Reaktion zu registrieren und zu messen. Dank dem mit der Coresign-Einheit des ARM Prozessors verbundenen Debug Access Port (DAP) kann, mit Hilfe des Serial Wire Debug Protokolls (SWD), der innere Zustand des Zielsystems sowie Simulators aufgenommen und in Echtzeit dargestellt werden. Mit diesen Mitteln kann die Messung anhand eines Traces nachvollzogen werden.

Die Messungen werden zeigen, ob und inwieweit sich die Ergebnisse des anwendungsbezogenen Benchmarks von denen des Rhealstone Benchmarks [2] unterscheiden und ob im Fall des untersuchten Open Source Betriebssystemes tatsächlich, wie erwartet, die Verwaltung der Prioritäten Hauptursache für den maximalen Verzug ist.

[1] Yiu, J. (2011). The Definitive Guide to the ARM Cortex-M0. Elsevier Science.

[2] Dr. Dobb's Journal: Rhealstone: a Real-Time Benchmarking Proposal

<http://www.drdoobs.com/rhealstone-a-real-time-benchmarking-prop/184408081> Abruf: 03.12.2015

[3] MiBench. Embedded Benchmark <http://www.eecs.umich.edu/mibench/> Abruf: 03.12.2015

[4] Gumzej, R. (2010). Real-time Systems' Quality of Service: Introducing Quality of Service. London: Springer.

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Eigene Darstellung

Konzeption und Realisierung eines Software-Moduls für den Test der Head-Unit eines Automobils mittels Restbussimulation auf Basis eines bewährten Test-Frameworks

Ali Celebi*, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Der kontinuierliche Wettbewerb, aber auch die sich ständig weiterentwickelnden Informationstechnologien, lösen Technologiesprünge in der Automobilindustrie aus. So ist das Auto heute nicht mehr einfach nur ein Fahrzeug auf vier Rädern, sondern ein fahrender Computer. Heute können Assistenzfunktionen, die mit Informationstechnologien realisiert sind, in einer kritischen Situation, ohne das Eingreifen des Fahrers, Gegenmaßnahmen einleiten. Aber auch die Infotainmentfunktionen, die die Fahrzeuginsassen unterhalten, informieren, sich wohlfühlen lassen und mit dem Rest der Welt verbinden sind ohne Informationstechnologien kaum umsetzbar. Umso wichtiger ist es, dass diese Technologien sehr zuverlässig funktionieren, da es in kritischen Situationen darauf ankommt die richtigen Maßnahmen einzuleiten, um mögliche Schäden zu vermeiden. Auch ist es von Bedeutung, dass das Entwicklungsstadium der jeweiligen Technologie so kurz wie möglich gehalten wird, weil sich die IT Branche bekanntermaßen sehr schnell weiterentwickelt und Technologien entsprechend schnell altern. Zum schnellen Vorantreiben ei-

ner Entwicklung gehört auch das Prüfen der eingesetzten Technologie auf mögliche Fehler, die während der Entwicklung von Hard- bzw. Software häufig entstehen. Hierzu werden unterschiedliche Verfahren angewandt. Ein Verfahren ist das funktionsorientierte Testen (Black-Box Test) der Hard- bzw. Software, die in den Fahrzeugen zum Einsatz kommen soll. Hierbei werden die Tests ohne den Kenntnisstand über die innere Funktionsweise des zu testenden Systems entwickelt.

Innerhalb der Telematikentwicklung der Firma Daimler wurde ein Test-Framework entwickelt, womit die in den Telematiksteuergeräten eingesetzte Software auf Fehler getestet werden kann. Ein Anwendungsfall für den Einsatz des Test-Frameworks ist die Simulation der Fahrzeugumgebung anhand einer sogenannten Restbussimulation. Dabei wird ein PC, auf dem das Test-Framework läuft, an den CAN-Bus angeschlossen, um der zu testenden Komponente, das Verhalten der restlichen Fahrzeugumgebung zu simulieren.

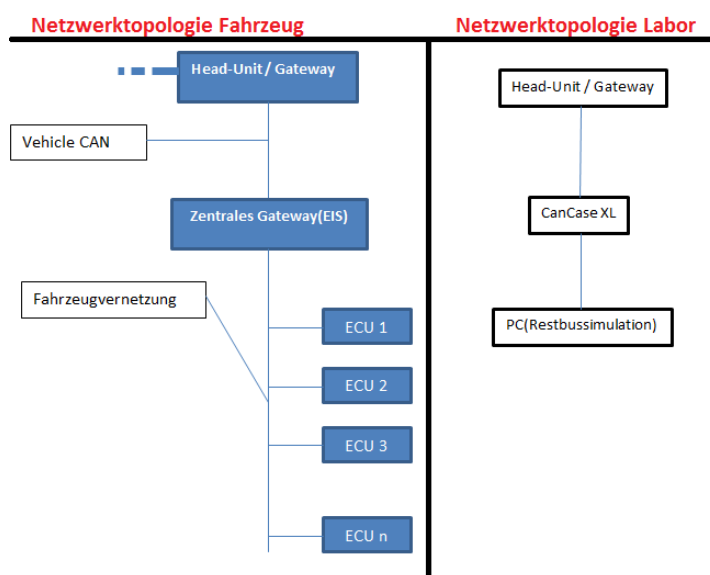


Abbildung 1: Links, schematische Netzwerktopologie im Fahrzeug und rechts, Netzwerktopologie in der simulierten Umgebung

*Diese Arbeit wurde durchgeführt bei der Firma Daimler AG, Sindelfingen

Der große Vorteil einer solchen Restbussimulation ist, dass nicht das ganze Fahrzeug, sondern nur die Komponenten benötigt werden, die getestet werden sollen. So kann man vom Labor aus Dauertests für die unterschiedlichsten Situationen durchführen, ohne sich dafür im Fahrzeug zu befinden oder gar Testfahrten machen zu müssen. Die Testjobs innerhalb des Test-Frameworks bieten dem Tester jedoch noch kein flexibles User Interface, können dafür aber recht schnell implementiert werden. Wenn ein flexibleres Interface benötigt wird, können für das Framework sogenannte Add-Ons geschrieben werden, bei denen der Gestaltung des Interfaces nahezu keine Grenzen gesetzt sind.

Die Bachelorarbeit befasst sich mit der

Kommunikation zwischen der Head-Unit und den Steuergeräten im Fahrzeug. Um zu überprüfen, ob das Kommunikationsverhalten den Spezifikationen entspricht und um mögliche Fehler im Verhalten der Head-Unit zu finden wird ein Add-On für das Test-Framework entwickelt. Dabei soll das Add-On die einzelnen Steuergeräte simulieren, die mit der Head-Unit kommunizieren. Hierzu wird eine Bedienoberfläche realisiert, bei der jedes Steuergerät der Simulation ein Interface zur Steuerung und Überwachung der Signale erhält. Die Steuergeräte selbst haben die unterschiedlichsten Aufgaben, wie z.B. die Klimaregelung, die Türverriegelung oder die Erzeugung von Warnsignalen bei Überschreitung eines Geschwindigkeitslimits.

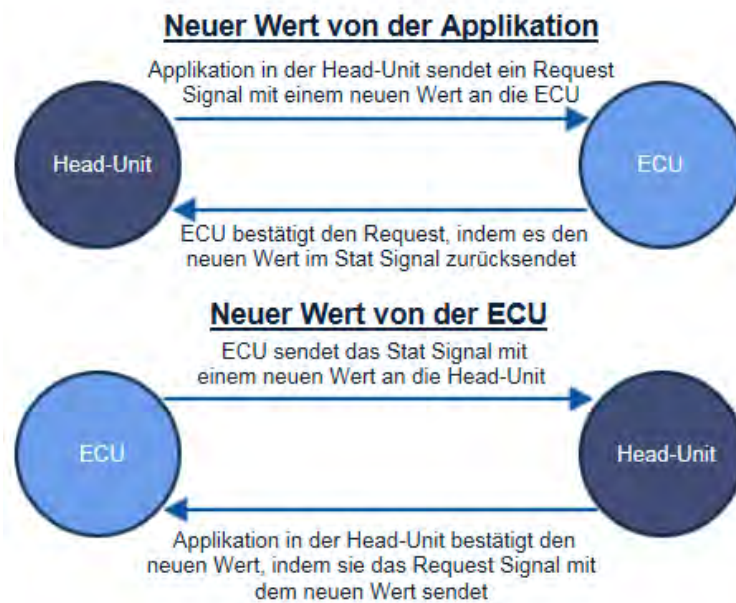


Abbildung 2: Handshake-Mechanismus - Jeweils ein Signal wird von der Head-Unit und vom Steuergerät verwendet um den Handshake-Mechanismus zu realisieren

Damit eine valide Kommunikation zwischen der Head-Unit und dem simulierten Steuergerät aufgebaut werden kann, wird auch ein Handshake-Mechanismus implementiert, der sicherstellt dass versendete Signale von der Gegenseite korrekt empfangen und akzeptiert wurden. Hierzu verwendet der Mechanismus zwei Signale. Ein Signal wird von der Head-Unit

an das Steuergerät versendet und das Signal, dass den Empfang eines korrekten Wertes bestätigt, wird vom Steuergerät an die Head-Unit zurückgesendet. Nun steht eine abgesicherte Datenübertragung zur Verfügung über den die Kommunikation der Head-Unit mit den Steuergeräten sichergestellt ist [1].

[1] Daimler AG: Doors (intern)

Bildquellen:

- Abbildung 1: Daimler AG
- Abbildung 2: Daimler AG

Integration eines Mikrocontroller-Motortreibers in den bürstenlosen Gleichstrommotor des Valeo Laserscanners „Scala 1403“

Tobias Claus*, Hermann Kull

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Fahrerassistenzsysteme wie ABS, ESP sind in Personenkraftwagen schon lange Standard, zunehmend werden aber auch Fahrerassistenzsysteme wie Abstandsregeltempomaten oder Spurhalteassistenten als Sonderausstattung eines Fahrzeuges angeboten. Diese Systeme sind erst durch eine kontinuierliche Weiterentwicklung der dazu benötigten Sensorik wirtschaftlich geworden. In der Vergangenheit sind insbesondere durch die Radar-Abstandssensorik in der Front des Fahrzeugs neue Funktionen wie der Abstandsregeltempomat in den PKW eingezogen. Mit der zusätzlichen Verwendung von Frontkameras hinter der Windschutzscheibe mit nachfolgender Bildverarbeitung wurde zum Beispiel auch die Erkennung der Fahrspuren und Verkehrsschilder möglich. Durch die Verwendung von beiden Technologien kann außerdem eine zentrale Sensordatenfusion erfolgen, bei der die Signale beziehungsweise Messergebnisse beider Systeme gegeneinander ausgewertet und zusammengeführt werden, womit die Genauigkeit der Messwerte im Fahrzeugumfeld deutlich gesteigert wird. Zusätzlich entsteht durch die Fusion eine Redundanz, die für Funktionen wie einen Notbremsassistenten dringend gefordert ist. Für zukünftige Systeme wie das autonome Fahren reicht diese zweifache Redundanz und Genauigkeit von den oben genannten Systemen nicht mehr aus, daher ist die Suche nach alternativen Sensorlösungen im Moment ein wichtiger Entwicklungszweig des zukünftigen Automobils.



Abbildung 1: Valeo Scala 1403

Aus diesem Grund entwickelt die Firma Valeo einen automobilen Laserscanner, der ähnlich funktioniert wie die bereits in der Industrieautomatisierung weit verbreiteten Geräte. Bei diesem Laserscanner kommt ein rotierender Ablenkspiegel zum Einsatz, der den Strahl des Lasers auf die Umgebung beziehungsweise das Fahrzeugumfeld ablenkt und somit die Messung von Entfernungen über den gesamten Öffnungswinkel möglich macht. Die Rotation des Ablenkspiegels wird durch einen bürstenlosen Gleichstrommotor bewirkt, der aktuell durch einen integrierten Schaltkreis in 6 Schritten sensorlos kommutiert wird. Dies kann Nachteile wie schwer zu kontrollierendes Anlaufen oder Drehmomentrippel mit sich bringen, da der Regler die Stellung des Motors nur durch den induzierten Gegenstrom in der nicht bestromten Motorwicklung messen kann.

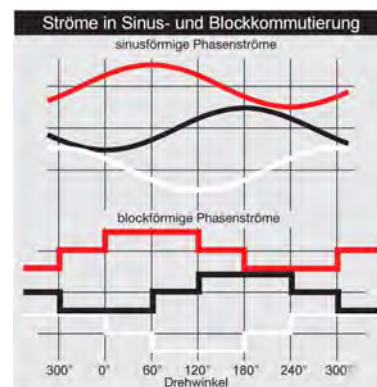


Abbildung 2: Vergleich der Ströme in Sinus- und Blockkommutierung

Dadurch entfällt die Möglichkeit einer Sinusbeziehungsweise Raumzeigermodulation, weil bei dieser alle Phasen fließend bestromt werden und daher nie eine freie Phase zur Messung der Rotorposition zur Verfügung steht. Zur Verbesserung der Ansteuerung soll der im Motor bereits integrierte Winkelsensor in Verbindung mit einem Mikrocontroller mit Raumzeiger-Modulations-Software zur Kommutierung des Motors genutzt werden. In der Bachelorarbeit wird dazu ein ARM Cortex-M3 Mikrocontroller mit bereits integrierter Leis-

*Diese Arbeit wurde durchgeführt bei der Firma Valeo Schalter und Sensoren GmbH, Bietigheim-Bissingen

tungselektronik zum direkten Anschluss der Motorspulen in die Leiterplatte des Motors auf sehr kompaktem Raum integriert. Daraus resultiert eine komplette Neuentwicklung der Leiterplatte mit den zum Betrieb des Mikrocontrollers benötigten Komponenten.

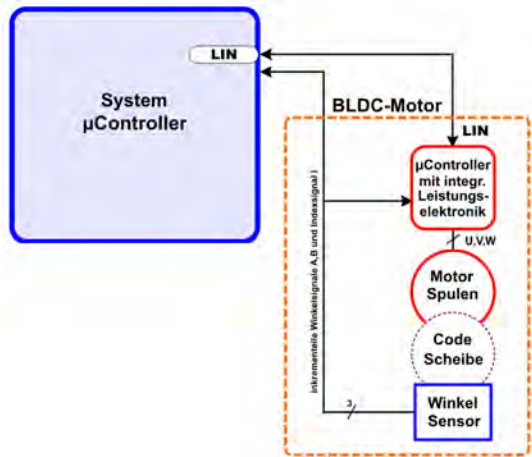


Abbildung 3: Blockdiagramm des Systemaufbaus

Die neuen Leiterplatten werden angefertigt und mit allen Komponenten bestückt, um damit lauffähige Prototypen des Motors aufbauen zu können. Desweiteren ist die Entwicklung der Software des Mikrocontrollers mit Hilfe von APIs des Mikrocontroller-Lieferanten ein weiterer Kernpunkt der Bachelorarbeit. In der Software des Motors wird die Kommutierung mittels einer Raumzeiger-Modulation auf Basis der Winkelsensor-Signale sowie ein PID-Regelungsalgorithmus zur Konstanthaltung der Motordrehzahl und somit der Drehzahl des Ablenkspiegels implementiert. Durch beim Fahrbetrieb auftretende Winkelbeschleunigungen wird der Ablenkspiegel außerdem abgebremst oder beschleunigt; darauf muss der Regelalgorithmus schnell reagieren um Störungen schnellstmöglich auszuregulieren und die Drehzahl im vorgegebenen Bereich zu halten. Eine weitere Anforderung ist ein möglichst schnelles Hochlaufen des Ablenkspiegels beim

Start des Laserscanners auf die vorgegebene Drehzahl. Als Rückführgröße des Reglers dienen ebenfalls die Signale des Winkelsensors, die durch eine Capture-Einheit des Mikrocontrollers gemessen werden um daraus die aktuelle Drehzahl zu berechnen.

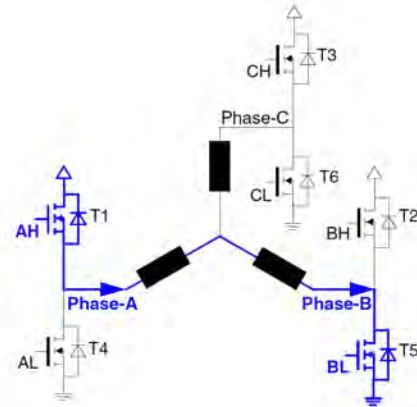


Abbildung 4: Leistungselektronik zur Ansteuerung der Leistungselektronik

Die Stellgröße des Reglers findet sich in Form eines PWM-Wertes wieder, der dem PWM-Ausgangswert der Raumzeigermodulation direkt überlagert wird. Zusätzlich zu den Inkrementalsignalen wird zur absoluten Positionsbestimmung des Motors pro Umdrehung ein Indexsignal durch den Winkelsensor generiert. Die Indexposition variiert von Motor zu Motor, was es notwendig macht in der Produktion jeden Motor einmalig zu kalibrieren. Die Kalibrierung geschieht in der Software des Mikrocontrollers, indem zunächst der Motor mit dem 0° -Raumzeiger bestromt wird. Dadurch springt der Motor in die 0° Position und wird per Zwangskommutierung langsam gedreht bis das Indexsignal gefunden wurde. Gleichzeitig werden die Signale des Winkelsensors ausgewertet und somit kann der Offset des Winkelsensor-Indexsignals relativ zur Motorposition genau bestimmt werden, was Voraussetzung für die Kommutierung des Motors ist.

- [1] Markus Maurer, Christoph Stiller – Fahrerassistenzsysteme mit maschineller Wahrnehmung
 [2] Micronas GmbH, Control of a Sensor-Based BLDC Motor using Six-Step Commutation
 [3] maxon motor gmbh, maxon EC motor, Technik – kurz und bündig

Bildquellen:

- Abbildung 1: Valeo Schalter und Sensoren GmbH
- Abbildung 2: maxon motor gmbh, maxon EC motor, Technik – kurz und bündig
- Abbildung 4: Micronas GmbH, Control of a Sensor-Based BLDC Motor using Six-Step Commutation

Design und Entwurf eines automatisierten Testsystems für Multiprotokoll-Sensoren im Kontext von Continuous Integration mit Definition eines applikationsspezifischen XML-Schemas

Marco D'Ottavio*, Walter Lindermeir

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Warum testet man überhaupt? Das Testen der Software dient der Qualitätssicherung und Qualitätsmessung. Die 10er Regel der Fehlerkosten zeigt die erheblichen Auswirkungen des Zeitpunkts der Fehlerbemerkung auf die Kostenentwicklung eines Projekts. Sie besagt, dass die Kosten für Findung und Behebung des Fehlers pro Projektphase um den Faktor 10 steigen [1]. Darüber hinaus wird das Risiko von Folgefehlern vermieden und eine Steigerung der Softwarequalität erreicht. In der Vergangenheit wurden Maßnahmen getroffen, um die Anzahl der Fehler zu reduzieren. Dazu zählen Standardisierung und neuartige Entwicklungsmethoden im Softwareentwicklungsprozess, wie beispielsweise modellbasierte Entwicklung [2]. Allerdings „hat die Fehlerrate pro 1000 Anweisungen [...] sich seit der Assembler-Programmierung nicht verringert“ [2]. Infolgedessen rückt das Testen immer mehr in den Vordergrund und nimmt einen Großteil der Arbeitszeit und somit auch der Kosten für ein Projekt in Beschlag. Das Unternehmen, für welches das Testsystem entwickelt wird, ist Entwickler von Sensoren. Aktuell wird das Testen zum Großteil manuell ausgeführt, was sehr aufwändig ist. Ausgangspunkt der Tests ist ein Dokument: die Testspezifikation, die Informationen wie:

- Name des Tests
- Beschreibung
- Autor, Abteilung
- Zweck/Ziel des Tests
- Vorbedingungen für den Test
- genaue Beschreibung der Testfälle
- Sollergebnisse

enthält. Im Anschluss werden die Aktionen der Testspezifikation von Hand durchgeführt und gleichzeitig ausgewertet und dokumentiert. Die Firmware für die unterschiedlichen Sensoren, die in der Abteilung entwickelt wird, basiert auf einer gemeinsamen Plattform. Dadurch lassen sich Auswirkungen auf die Funktionalität, durch Aktualisierungen und Fehlerbehebungen geräteübergreifender Softwarekomponenten, nur schwer vorherbestimmen

und abschätzen. Der Variantenreichtum in Bezug auf Geräte-, Funktionalitäts- und Schnittstellenvarianten der Produkte verschärft das Problem. Die Tests sollten daher nach jeder Änderung der Firmware durchgeführt werden, um die korrekte Funktionalität zu gewährleisten. Das bedeutet einen enormen Zeitaufwand. Zugleich steigt der Aufwand stetig durch immer mehr Funktionalität und Varianten, die hinzugefügt werden. Als Lösung für dieses Problem wird eine Automatisierung für den gesamten Prozess vorgenommen.

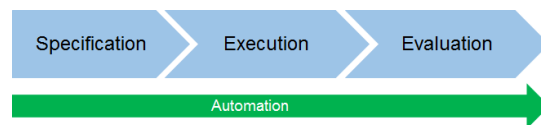


Abbildung 1: Der Prozess des Testens

Diese hat neben der offensichtlichen Zeiterparnis und der damit einhergehenden Geldersparnis weitere bedeutende Vorteile. Durch die Unabhängigkeit des Tests von der Aufmerksamkeit eines Testers lassen sie sich problemlos nachts ohne menschliche Beteiligung durchführen. Dementsprechend besteht eine ständige Rückkopplung an den Entwickler. Durch die Automatisierung ist es außerdem möglich, exakte Wiederholungen durchzuführen. Ein weiterer Vorteil ist die Möglichkeit der Parallelisierung der Tests ohne personellen Mehraufwand. Das gesamte Projekt steht unter dem Namen „Continuous Integration“(CI). Der Begriff beschreibt das Zusammenstellen der Software aus einer gemeinsamen Codebasis, die aus unterschiedlichen unabhängigen Komponenten besteht und den anschließenden Tests. Wobei dies Komponententests, Integrationstests oder wie in diesem Fall Systemtests sein können. Die Integration wird auf einem separaten CI-Server ausgeführt. Als Software-System wird Jenkins verwendet. In Abbildung 2 ist die Laborumgebung dargestellt.

* Diese Arbeit wurde durchgeführt bei der Firma Leuze electronic GmbH & Co. KG, Owen

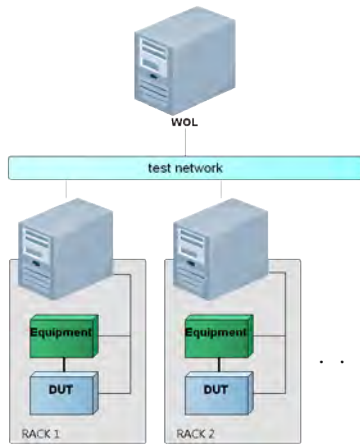


Abbildung 2: Die Laborumgebung

Der Wake-On-LAN-Rechner (WOL-Rechner), der ständig eingeschaltet ist, dient als Ansprechpartner für den CI-Server. Die Racks enthalten einen Standard-PC und eine spezifische Testumgebung. Als Hilfsmittel stehen jedem Rack ein Netzteil, Monitore, Relaiskarten zum Ein/Ausschalten der Sensoren oder zum Triggern verschiedener Eingänge und Dateneingangskarten zur Kontrolle von Ausgängen der Sensoren zur Verfügung. Alle Geräte des Racks sind an eine Master-Slave-Steckdose angeschlossen, wobei der Rechner an der Master-Steckdose angeschlossen ist. Nachdem Jenkins eine neue Version der Firmware fertiggestellt hat, wird als Post-Build-Action die Steuerungssoftware auf dem WOL-Rechner gestartet. Diese übernimmt nun das Starten der Rack-PCs und die Verteilung auf die entsprechenden Racks, die Initialisierung der Geräte, das Aktualisieren des Devices Under Test (DUT), die Testdurchführung und die Dokumentation der Ergebnisse. Am Ende des Tests erhält Jenkins Berichte im JUnit-Format

über den Testverlauf und anschließend werden die Racks abgeschaltet, um die Lebensdauer der PCs zu verlängern und unnötigen Stromverbrauch zu vermeiden. Das automatisierte Testsystem besteht im Grunde aus zwei Bestandteilen: einem XML-Schema und einer Software zum Verteilen und Steuern der Tests. Für jede neue Baureihe muss ein XML-Dokument erstellt werden. Mit Hilfe des Schemas kann das XML-Dokument bei der Erstellung validiert werden und sichergestellt werden, dass eine gültige Testbeschreibung entsteht. Die Baureihe des Sensors dient als Wurzelement des XMLs. Neben allgemeinen Informationen zur Serie werden hier die Racks bestimmt, auf denen Tests mit der Baureihe ausgeführt werden sollen. Ein Rack ist auf eine Baureihe beschränkt, allerdings können Tests für eine Baureihe auf mehrere Racks verteilt werden, damit können umfangreiche Tests parallelisiert werden. Das Rack-Element beinhaltet die Hilfsgeräte, die vorhanden sind und die Definition eines oder mehrerer DUTs. Das DUT wiederum enthält Informationen über die vorhandenen Schnittstellen des Geräts und die zu verwendenden Protokollvariationen. Darunter beginnt die eigentliche Beschreibung der Testspezifikation und den Testfällen, die Aktionen und die erwarteten Reaktionen des Tests beschreiben. Aktionen und Reaktionen, die mit Hilfe eines Zusatzgeräts oder mit einer vorher definierten Schnittstelle ausgeführt werden, erhalten einen Verweis auf eine Ressource, sodass nur Schnittstellen und Geräte verwendet werden, die auch vorhanden sind. Die Steuerungssoftware bearbeitet die Informationen dieser XML-Beschreibung. Ein besonderes Augenmerk liegt auf der Erweiterbarkeit des Testkonzepts, so sollen neue Geräte und Schnittstellen mit minimalem Aufwand integrierbar sein.

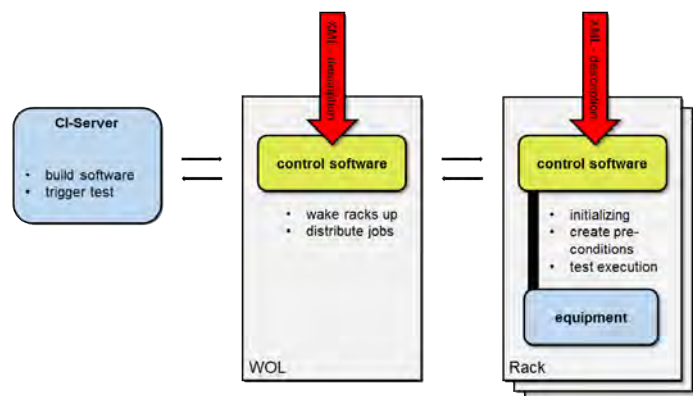


Abbildung 3: Das Testsystem

[1] Mellis, Werner: Projektmanagement der SW-Entwicklung, 1. Auflage. Wiesbaden, S.198

[2] Sneed, Harry M. / Baumgartner, Manfred / Seidl, Richard (2009): Der Systemtest. 2. Auflage. München, S. 7

Bildquellen:

- Abbildung 1-3: Eigene Darstellung

Untersuchungen zur Entwicklung einer Diagnosestrategie für Smart NO_x-Sensoren im Fahrzeugbetrieb

Fayez Daas*, Hermann Kull, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Emissionen von Kraftfahrzeugen spielen gegenwärtig eine wichtige Rolle. Die Automobilhersteller müssen entsprechende Emissions-Limits einhalten, um Kraftfahrzeuge zulassen zu können. Um die Grenzwerte der Emissionen einhalten zu können, werden zum Teil inner-motorische Maßnahmen durchgeführt. Über das Motorsteuergerät wird die optimal benötigte Kraftstoffmenge zur Frischluftmenge im Zylinder ermittelt und eingespritzt. Aufgrund der verschärften Emissionslimits in den letzten Jahren sind zusätzliche Entwicklungen und Maßnahmen notwendig. Einer dieser Maßnahmen ist die Abgasnachbehandlung. Demzufolge ist auch ein Abgasnachbehandlungssystem erforderlich, um die, vom Verbrennungsmotor emittierten Schadstoffe, zu filtern bzw. chemisch in unschädliche Stoffe zu konvertieren. Beim Benzinmotor agiert der allgemein bekannte drei-Wege-Katalysator als Abgasnachbehandlungssystem. Allerdings ist die Abgasnachbehandlung beim Dieselmotor

weitaus komplexer und besteht aus mehreren Katalysatoren. Die Komplexität der Abgasnachbehandlung nimmt mit neuen Emissionsrichtlinien weiterhin zu. Um die, bei der Verbrennung entstehenden Rußpartikel zu filtern, wird ein Partikelfilter verwendet. Ein Diesel-Oxidationskatalysator (DOC) wird für die Reduktion der HC- und CO-Emissionen eingesetzt. Die Reduktion der Stickoxide erfolgt durch einen anderen Katalysator. Dies kann beispielsweise ein NO_x-Speicherkatalysator sein, der die Stickoxide speichert und chemisch in Stickstoff und Sauerstoff umwandelt oder aber auch ein SCR-Katalysator (Selektive chemische Reduktion), der mittels Zugabe von Harnstoff (bekannt auch als AdBlue) die Stickoxide im Abgassystem chemisch reduziert. In der Abbildung 1 ist ein Abgasnachbehandlungssystem eines Dieselmotors abgebildet. Dieses besteht aus einem Oxidationskatalysator, einen NO_x-Speicherkatalysator, ein Partikelfilter und diverse Abgassensoren.

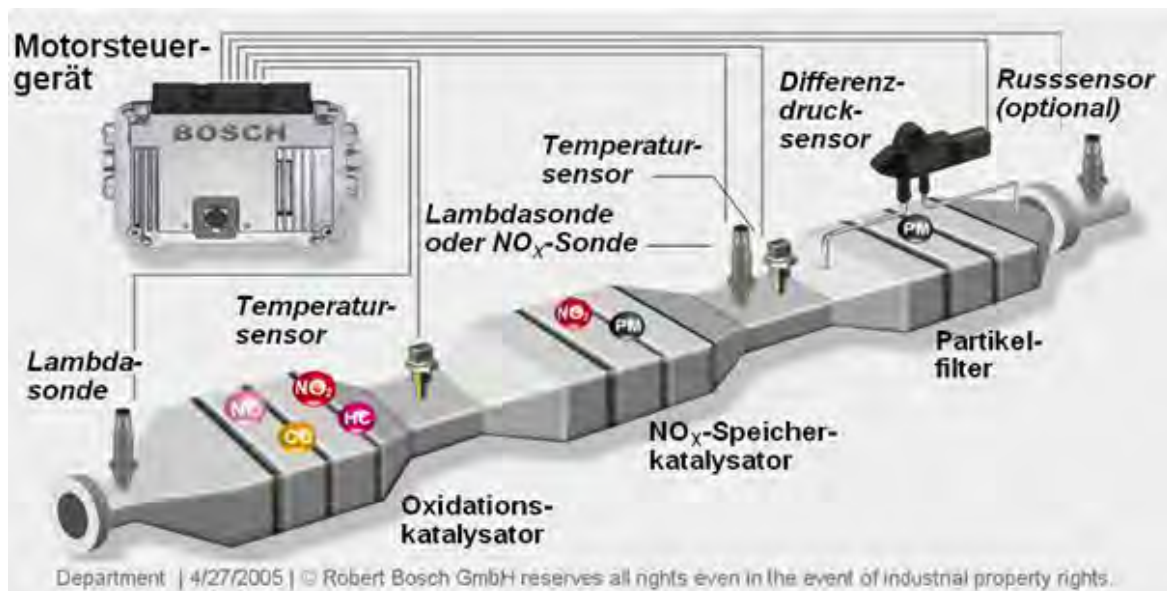


Abbildung 1: Diesel Abgasnachbehandlungssystem

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Stuttgart-Feuerbach

Damit die Katalysatoren und Filter überwacht werden können und einwandfrei funktionieren, werden Abgassensoren eingesetzt. Dabei spielt der NO_x-Sensor eine zentrale Rolle, denn durch die Erfassung der Stickoxid-Anteile ist dieser sehr essentiell. Jede Abgasnachbehandlungskomponente hat ihre eigene Funktion und im Zusammenspiel erfüllen diese die notwendigen Grenzwerte. Zudem müssen laut OBD-II-Norm (On Board Diagnose II), alle abgasbeeinflussenden Systeme während des Fahrbetriebs überwacht werden und bei einer Fehlfunktion dem Fahrer über eine Motorkontrollleuchte gemeldet werden [1].

Im Rahmen dieser Abschlussarbeit soll für einen NO_x-Sensor ein Verfahren zur Auswertung der Signaldynamik entwickelt werden. Um die Signaldynamik des NO_x-Sensors auszuwerten, wurde der Sprung im Signalverlauf mit

unterschiedlichen mathematischen Verfahren untersucht und miteinander verglichen.

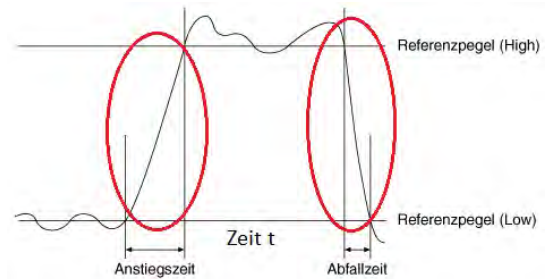


Abbildung 2: Sprung im Signalverlauf zum auswerten

Ziel dieser Abschlussarbeit ist es, ein Verfahren zu entwickeln, mit dem eine Veränderung der Signaldynamik im realen Fahrbetrieb am besten erkannt werden kann.



Abbildung 3: Abgasnachbehandlung NO_x-Sensor

[1] Konrad Reif(Hrsg.) (2012): Dieselmotor-Management | Vieweg+Tuebner Verlag, 5. Auflage

Bildquellen:

- Abbildung 1: <http://www.icvt.uni-stuttgart.de/bilder/projekte/abgas2.png>
- Abbildung 2: Eigene Darstellung
- Abbildung 3: http://produkte.bosch-mobility-solutions.de/media/ubk_europe/db_application/downloads/pdf/antrieb/de_5/150909_Bosch_DS_NOxSensor-EGS-NX_DE.pdf

Konzeption und Design einer Webanwendung zur BAföG-Berechnung für Antragsteller und Sachbearbeiter

Tobias Dahmen*, Peter Väterlein, Harald Melcher

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Entwicklung von Webanwendungen durchlebt einen starken Wechsel, weg vom Server, hin zum Client. Durch die zunehmende Leistung der Endgeräte werden immer mehr Berechnungen und Funktionalitäten auf diese ausgelagert. Hierdurch wird der Server entlastet und der Anwender erhält eine schnell reagierende Anwendung.

Serverseitige Technologien wie ASP.Net werden durch JavaScript ersetzt und dies durch Frameworks unterstützt, um nicht jedes Mal das Rad neu zu erfinden. Genau das stellt den Entwickler vor eine Entscheidung.



Abbildung 1: JavaScript-Frameworks Logos

JavaScript-Frameworks gibt es wie Sand am Meer, da sich viele Entwickler die Aufteilung und Verknüpfung der Anwendung anders vorstellen und unterschiedliche Zusatzfunktionen bevorzugen. Somit gab es im Laufe der letzten Jahre immer wieder Neuentwicklungen, die von Unternehmen wie Google oder Facebook, von großen Communities oder von wenigen Entwicklern entwickelt wurden. AngularJS, Aurelia, Backbone, Ember, Handlebars, Knockout, Meteor, Polymer und React sind nur eine Auswahl der Frameworks, die aktiv weiterentwickelt werden. Welches Framework für welche Aufgabe das „richtige“ ist, hängt nicht nur von den Eigenheiten der Frameworks, sondern auch von persönlichen Präferenzen sowie der, teils recht kurzen, Lebenszeit der Frameworks ab. So gibt es Frameworks die auf maßgeschneiderten HTML-Elementen aufbau-

en, Frameworks die die Verknüpfung der Daten an die Ansicht mit Änderungen in beiden Richtungen unterstützen und Frameworks bei denen sich die HTML-Ausgabe direkt über JavaScript beeinflussen lässt.

Unterschiede, die sich deutlich auf die Programmstruktur auswirken, gibt es vor allem bei der Verknüpfung von den Daten mit der Ansicht, dem sogenannten Databinding. AngularJS zum Beispiel, bevorzugt hierfür *dirty-checking*, dabei werden die gebundenen Objekte regelmäßig auf Änderungen überprüft, um die Ansicht gegebenenfalls anzupassen. Dies ermöglicht eine Programmstruktur, bei der die gebundenen Objekte wie gewohnt gesetzt werden können, ohne spezielle Richtlinien zu beachten. Bei Ember hingegen müssen spezielle *set*-Methoden genutzt werden.

Hierbei gibt es nicht nur Unterschiede in der Struktur, sondern auch bei der Performance. *Dirty-checking* wirkt sich im Normalfall, selbst bei Datenmengen von mehreren Zehntausend Parametern nicht negativ auf die Performance aus, dies bleibt selbst bei vielen parallelen Änderungen konstant. Jedoch ist zu beachten, dass bei komplexen Datenbindungen der Browser schon bei wenigen Tausend merklich langsam werden kann. Ein weiterer Flaschenhals bleibt die Erstellung der Elemente, da hier der HTML-Code verändert werden muss, so kann das Erstellen von mehreren Tausend Elementen, den Browser an seine Grenzen bringen. Das Gegenteil findet man bei *set*-Methoden, hier beeinflusst die Datenmenge oder die Komplexität der Bindungen die Performance nicht signifikant, doch wirken sich viele parallele Änderungen negativ aus [1]. Wichtig ist, dass ein Wechsel der Frameworks oft einen großen Mehraufwand darstellt. Hierbei ändern sich Berechnungsroutinen und Funktionen meist wenig, da es sich überall um JavaScript handelt. Jedoch gibt es strukturelle Unterschiede vor allem bei den HTML-Templates der verschiedenen Frameworks. React zum Beispiel setzt hier auf JSX, eine Erweiterung von ECMAScript (JavaScript ist eine Art Dialekt von ECMAScript), damit ist es möglich im JavaScript-Code XML ähnlichen Syntax zu verwenden, um damit HTML-Elemente zu er-

*Diese Arbeit wurde durchgeführt bei der Firma Datagroup IT Solutions GmbH, Pliezhausen

zeugen. Die meisten anderen Frameworks setzen auf HTML-Templates die durch zusätzliche HTML-Elemente oder Speziellen Syntax, wie `${<code>}` oder `{{<code>}}`, ergänzt werden. Die Unterschiede der Datenmodelle, an die gebunden werden kann, sind ebenso meist sehr groß, zusätzlich verwenden die HTML-

Templates teils komplett andere Logiken. So werden Listen in Ember und Handlebars mithilfe des `{{#each}}` Helpers dargestellt, wobei in React die Elemente mithilfe der `Array.prototype.map()` Funktion erstellt werden (Abbildung 2) in AngularJS hingegen wird hierfür die `ng-repeat` Direktive genutzt.

<pre> 1 <script type="text/x-handlebars"> 2 {{input type="text" 3 value=todotext}} 4 <button {{action "add"}} 5 value="Add"/> 6 7 {{#each todo in todos}} 8 9 {{todo}} 10 <button {{action 11 "remove" todo}} 12 value="X"/> 13 14 {{else}} 15 Keine Eintraege 16 {{/each}} 17 18 </script> </pre>	<pre> 1 return (2 <div> 3 <input type="text" value={ 4 todotext} onChange={this. 5 handleChange} /> 6 <button onClick={this.add} 7 value="Add"/> 8 9 {todos.length? 10 todos.map(function(todo) { 11 return {todo} 12 <button onClick= 13 {remove.bind(this,todo)} 14 value="X"/> 15 ; 16 }) : 'Keine Eintraege'} 17 18 </div> </pre>
--	---

Abbildung 2: Codevergleich Ember (links) und React (rechts)

Die Möglichkeit, Templates in separate HTML-Dateien auszulagern, wird von den meisten Frameworks geboten, oft bedingt durch die dezentrale Struktur von HTML.

Das Projekt hat eine Webanwendung zum Ziel, die Berechnungen auf relativ wenig Daten (Parameterzahl < 100) durchführt, so spielt die Performance der Datenbindung keine Rolle. Ein großer Faktor der Anwendung ist die dynamische Eingabehilfe, die dem Anwender unnötige eingaben ersparen soll. Dafür müssen viele Teile der Anwendung verknüpft werden. Hierdurch eignet sich die Struktur der meisten Frameworks nicht, da diese die Daten nur an untergeordnete Elemente weiterreichen, um sie darzustellen. Somit müsste die komplet-

te Anwendung in einem Element untergebracht werden, oder diese mithilfe unkonventioneller Methoden aufgeteilt werden.

AngularJS bietet die Möglichkeit Anwendungen beliebig aufzuteilen, hier steht aber noch offen, wie lange dieses Framework unterstützt wird, da die erste stabile Version von Angular 2 im ersten Quartal 2016 veröffentlicht werden soll. Angular 2 eignet sich weniger für die Anwendung, da sich die Struktur mehr an die der meisten anderen Frameworks angleicht. Der offizielle Blogbeitrag, dass das Angular 1 so lange unterstützt wird, wie es von der Mehrheit der Entwickler verwendet wird [2], gibt hier keine konkrete Angabe.

[1] <https://angularjs.de/buch/dirty-checking>

[2] <http://angularjs.blogspot.de/2015/11/highlights-from-angularconnect-2015.html>

Bildquellen:

- Abbildung 1: Tobias Dahmen
- Abbildung 2: Tobias Dahmen

Konzipierung und Implementierung einer auf dem Client-Server-Modell basierenden Kommunikationsschicht zur Erweiterung einer generischen Plattform für die Entwicklung von Automatisierungskomponenten mit dem Ziel zeitgleicher logischer Verbindungen mehrerer Applikationen zu einem Feldgerät

Siegfried Derksen*, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Heutzutage werden Aktuatoren und Sensoren vermehrt über Feldbussysteme zur Ansteuerung von Industrieanlagen und Maschinen eingesetzt. Die Vorteile der Feldbussysteme liegen in einer Minimierung des Verdrahtungsaufwands, einer Verringerung der Ausfallrate der einzelnen Komponenten und im Erreichen eines hohen Automatisierungsgrads [1]. Die Kommunikation mit den Feldgeräten kann aber auch schon während der Entwicklung zur Analyse und Überwachung der Prozessparameter eingesetzt werden. Für einen hohen Grad der Softwarequalität und der Zuverlässigkeit müssen während der Entwicklung verschiedene Tests durchgeführt werden. Dabei ist es von Vorteil, wenn die Prozessparameter nach außen sichtbar sind und diese gesetzt und abgefragt werden können. Der Zugriff auf die Feldgeräte von einem entfernten Objekt wird dann mithilfe eines Remote-Services realisiert. Somit können Parameter während der Laufzeit ausgelesen und gesetzt werden, ohne dass hierfür eine neue Feldgerätesoftware implementiert werden muss. Diese Funktionalitäten werden in einer Middleware zur Verfügung gestellt.

Ein Gesamtsystem kann dazu in drei Schichten zusammengefasst werden: Anwendungsschicht, Betriebssystemschicht und Hardware schicht. Die Middleware befindet sich als vierte Schicht zwischen der Applikations- und Betriebssystemschicht [2].

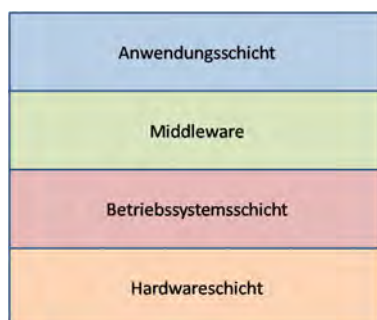


Abbildung 1: Schichtenmodell mit Middleware

Die Hauptaufgabe einer Middleware ist die Vereinfachung der Kommunikation mit den unteren Schichten und das Verbergen der Komplexität der Softwarestruktur [2]. Die zusätzliche Schicht kann von einer PC-Applikation genutzt werden, um eine Verbindung per Remote-Service herzustellen. Dabei wird eine Peer-to-Peer-Verbindung aufgebaut. Der große Nachteil dieser Verbindung besteht darin, dass zeitgleich nur eine Applikation mit dem Feldgerät kommunizieren kann [3].

Im Rahmen dieser Bachelorarbeit soll auf Basis der Middleware ein Interface definiert und implementiert werden, der den Zugriff des Remote-Service zeitgleich für mehrere Applikationen erlaubt. Diese Art der Kommunikation stellt eine Client-Server-Beziehung dar. Die Applikationen repräsentieren hierbei die Clients. Unter dem Interface wird auf Basis der Middleware ein Server implementiert.

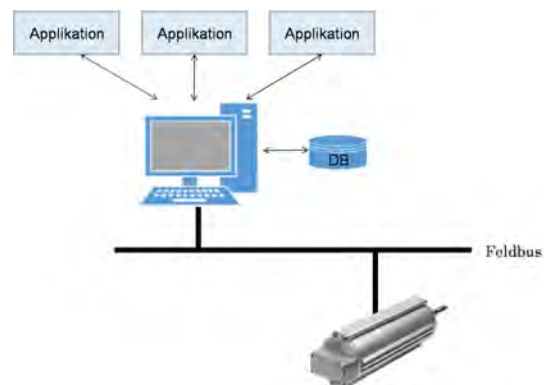


Abbildung 2: Client-Server-Kommunikation

Bevor ein Datenaustausch zwischen Applikationen und Feldgerät gestartet werden kann, benötigt der Server zusätzliche Informationen über die Kommunikationspartner.

Zu Beginn der Kommunikation wird eine Modelldescription vom Feldgerät geladen. Darin wird der ganze Funktionsumfang abgebildet, der von außen zur Verfügung steht. So-

*Diese Arbeit wurde durchgeführt bei der Firma Festo AG & Co. KG, Denkendorf

mit können azyklische Informationen zur Parametrisierung in das Feldgerät geschrieben werden. Wenn eine Applikation eine Funktion oder einen Parameter ausführen oder ändern will, muss dieses Objekt auf dem Feldgerät ausgeführt werden.

Die Middleware stellt Funktionen und Interfaces zur Verfügung, die von den Applikationen genutzt werden können.

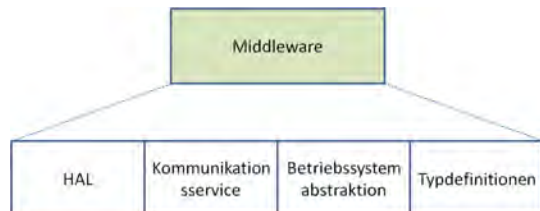


Abbildung 3: Schnittstellen der Middleware

Mithilfe des LUA-Interpreters können Objekte während der Laufzeit erzeugt und ausgeführt werden. LUA ist eine Skriptsprache, die für ihre leistungsfähigen Skripte und einen schlanken Interpreter bekannt ist [4]. Somit können die Objekte von außen angesprochen und ausgeführt werden. Da das System generisch aufgebaut ist und die Informationen während der Laufzeit ermittelt werden, hat der Server keinen spezifischen Bezug zum Feldgerät.

Mithilfe des Interfaces können Parameter und Funktionen im Push-und-Pull-Verfahren gesetzt und abgefragt werden. Dabei wird von der Applikation entschieden, ob die Parameter zyklisch oder azyklisch aktualisiert werden. Bei einem Pull-Verfahren muss der Empfänger aktiv nachfragen, ob der beobachtete Parameter verändert wurde. Bei der Push-Variante wird der Empfänger vom Server benachrichtigt, sobald der beobachtete Parameter aktualisiert wurde. Bei zyklischen Aufrufen werden die Ergebnisse, die vom Feldgerät geladen werden, in eine Datenbank abgelegt. Es muss darauf geachtet werden, dass der mehrfache Zugriff auf die Datenbank ausgeschlossen ist. Die Feldgeräteinformationen werden als String an den Server zurückgeliefert. Darin befinden sich Informationen über die Objekte, die Methoden, die Typen der Funktionsparameter und der Rückgabebetyp. Zwischen Server und Feldgerät werden Daten als Strings ausgetauscht. Darin werden die Objekte, Methoden und Werte als String übergeben. Dies wird vom LUA-Interpreter verarbeitet und zu den Kommunikationspartnern weitergeleitet. Aus der Sicht der Applikationen sollen diese Methoden mit den Datentypen aufgerufen werden, die auch im Feldgerät definiert sind. Da die Anzahl der Funktionsparameter variiert, muss das Interface diese Variabilität im Methodenaufbau unterstützen.

-
- [1] Frequenz Elektro GmbH: Feldebussysteme – <http://www.frequenzelektro.de/prozessautomation/feldebussysteme.html> Zugriff: 25.11.2015
 [2] DATACOM Buchverlag GmbH: Middleware – <http://www.itwissen.info/definition/lexikon/Middleware-middleware.html> Zugriff: 25.11.2015
 [3] Festo AG & Co. KG: Pflichtenheft – interne Quelle
 [4] LabLua: About LUA – <http://www.lua.org/about.html> Zugriff: 30.11.2015

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Festo AG & Co. KG
- Abbildung 3: Eigene Darstellung

Konzeption und Realisierung einer Anbindung von Mikrocontroller-basierter Hardware an mobiles Multimediagerät zur Datenvisualisierung

Dharyl Christ Djigoue Pouani*, Reiner Marchthaler, Werner Zimmermann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Das Elektroauto hat das Potenzial, in der nahen Zukunft das Auto mit herkömmlichem Verbrennungsmotor zu übertreffen. Neben den schon sehr umweltfreundlichen Hybrid-Autos sind Elektroautos ein emissionsloses und sehr geräuscharmes Fahren. Die Firma comemso verfügt über eigene Produkte im Bereich E-Mobility. Hierzu stehen der hochpräzise Batteriezellen-Simulator für die Entwicklung und Validierung moderner Batterie-Management-Systeme (BMS) von Elektro- und Hybridfahrzeugen zur Verfügung. Die comemso GmbH ist der innovative Spezialist für kundenspezifische embedded Systeme mit den Anwendungsschwerpunkten Automotive, E-Mobility und E-Mobility-Testing [1].

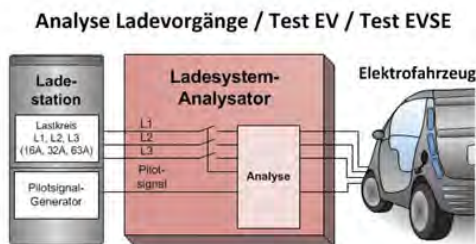


Abbildung 1: Ladesystem-Analysator

In der angefertigten Bachelorarbeit wurde einer Anbindung von Mikrocontroller-basierter Hardware an mobiles Multimediagerät zur Datenvisualisierung realisiert. Diese ermöglicht dem Kunden den Ladevorgang mit einem Smartphone/Tablet zu überwachen. Die Datenkommunikation erfolgt über ein ausgewähltes Bluetooth-Modul mit ungefähr 100 Meter Datenübertragung. Darüber hinaus wurde eine Android-Applikation für die Daten Visualisierung entwickelt [2]. Bei allen Maßnahmen, besonders bei der Erstellung der Apps-Design, wurde eine Erweiterungsmöglichkeit berücksichtigt, damit es die Möglichkeit für eine andere Nutzung oder eine Erweiterung gibt.



Abbildung 2: Darstellungskonzept

Allerdings ist es auch wichtig für die Implementierung der Applikation auf Android Betriebssystem einige Konfiguration zu erstellen. Über eine spezielle Sonderfunktion lassen sich die Entwicklungsoptionen öffnen, wo die Option USB-Debugging aktiviert werden soll. Diese Option ist für die Kommunikation des Android-Geräts mit PC wichtig. Damit können mit den ADB-Tools bestimmte Befehle ausgeführt werden und beispielsweise können das Systemlogbuch angezeigt oder Dateien gelöscht, verändert oder erstellt werden. Darüber hinaus sollte die USB-Debugging-Funktion eingeschaltet werden, nur wenn ein zwingender Bedarf vorhanden ist und nach dem Erledigen der Aufgaben wieder ausgeschaltet werden. Die Implementierung des Programms zur Entwurf auf dem Mikrocontroller erfolgt in C-Programmiersprache. Zuerst werden Ports von Bluetooth und Mikrocontroller initialisiert und während des Ablaufs des Programms, wird immer überprüft, ob die von Bluetooth Kommenden Daten für

*Diese Arbeit wurde durchgeführt bei der Firma comemso GmbH, Ostfildern

das jeweilige Display übereinstimmen. Bluetooth Low Energy (BLE) hat verschiedene Rollen in Abhängigkeit von der Art der Verbindung. In verbindungsloser Übertragung sind diese „Broadcaster“ und „Observer“, für die Verbindungsorientierte Übertragung „Peripheral“ und „Central“. Für die Kommunikation im verbindungsorientierten Modus arbeitet der BLE entweder als Master oder Slave und unterstützt keinen Master-Slave Switch. Zur Daten Sicherheit verwendet Bluetooth Low Energy den „AES-128 Block Cipher and Counter with Cipher Block Chaining Message Authentication Code“. Alle für die Sicherheit benötigten Schlüssel werden bei Bluetooth Low Energy auf dem Host generiert und dieser Ansatz hat geringe Rechenleistungs- und Speicheranforderungen und ermöglicht einen preisgüns-

tigen Aufbau des Controllers. Verschlüsselung kann vom Peripheral oder Central Device gestartet werden und für jedes Paket wird ein Message Integrity Check durchgeführt. Für die Kommunikation verwendet BLE das Standardkonzept „Client/Server-Modell“ das heißt der Client sendet einen Request für die Abfrage von einem oder mehreren Attribute an den Server und dies bietet die Attribute mit Hilfe der Protokoll-Attribute (ATT). Im Verbindungsorientierten Mode gibt Privacy einen Sicherheitsvorteil. Damit ist es schwer für einen Angreifer (Attacker) die Kommunikation zwischen den Geräten (Devices) zu verfolgen. Geräte mit aktivierter Privacy ändern mindestens alle 15 Minuten die Bluetooth-Adresse. Privacy wird auch für erneute Verbindungen verwendet.

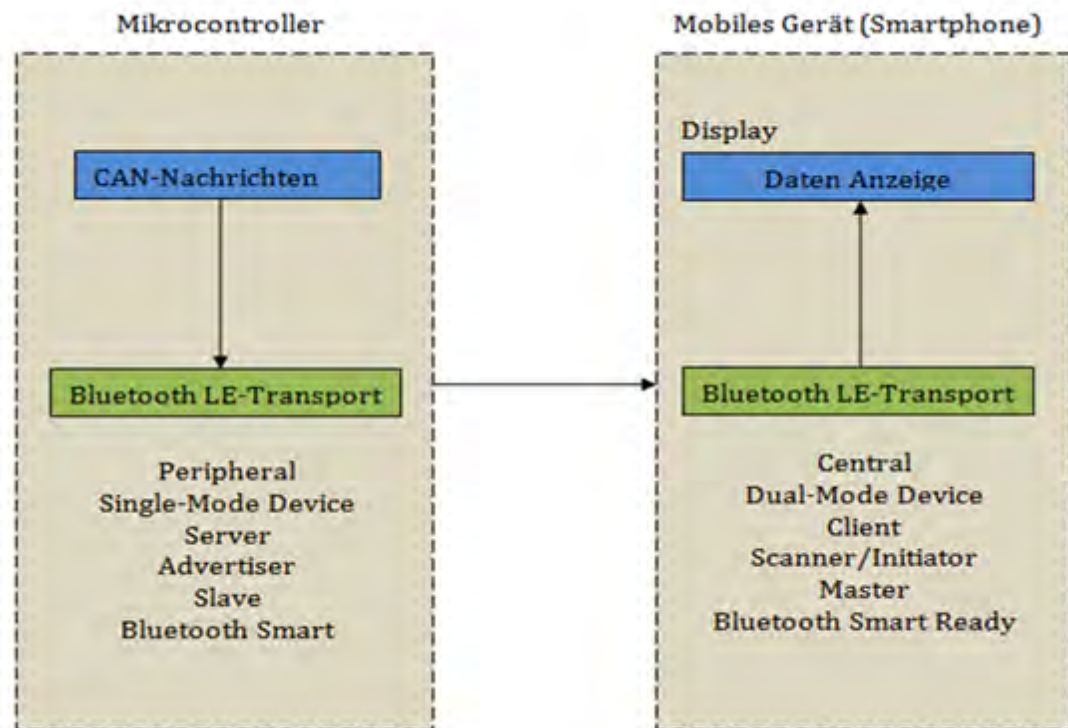


Abbildung 3: BLE Device-Rollen

[1] <http://www.comemso.de/index.php/de/produkte/produkte-uebersicht>.

[2] comemso GmbH, „interne Quelle: Bachelorarbeit,“ Talstraße 49-51, D-73760 Ostfildern/Germany, 2016.

Bildquellen:

- Abbildung 1: <http://www.comemso.de/index.php/de/produkte/produkte-uebersicht>
- Abbildung 2: comemso GmbH, „interne Quelle: Bachelorarbeit,“ Talstraße 49-51, D-73760 Ostfildern/Germany, 2016.
- Abbildung 3: -

Fehlertolerante Systeme durch effektive Laufzeitdiagnose mit SFL

Philipp Dürnay*, Hans-Gerhard Groß

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Fehlertolerante Systeme haben die Fähigkeit, trotz einer begrenzten Anzahl fehlerhafter Subsysteme ihre spezifizierte Funktion weitestgehend zu erfüllen [1]. Damit wird die Zuverlässigkeit des Systems erhöht. Fehlertoleranz spielt somit eine wichtige Rolle für autonome und sicherheitskritische Systeme.

Spectrum-based Fault Localization (SFL) ist ein Verfahren zur Fehlerlokalisierung und eignet sich aufgrund seiner geringen Ressourcenanforderung für den Einsatz zur Laufzeit. Eine Herausforderung des Verfahrens bleibt jedoch der Overhead, der durch die Sammlung der Daten (**Monitoring**) entsteht. In diesem Artikel wird das Verfahren SFL im Rahmen von fehlertoleranten Systemen vorgestellt. Dazu wird zunächst der Begriff des Fehlers erläutert und eine Übersicht über fehlertolerante Systeme gegeben, bevor genauer auf das Verfahren SFL eingegangen wird. Im Anschluss wird ein Ansatz geschildert effizienteres Monitoring mithilfe von Publish/Subscribe-Systemen zu realisieren.

1 Grundlagen

Ein Fehler besteht nach Laprie [1] aus drei Aspekten:

- **Fehlerursache** (eng. fault)
Die Fehlerursache ist der eigentliche Grund für den Fehler wie zum Beispiel eine falsch programmierte Anweisung.
- **Fehlzustand** (eng. error)
Der Fehlzustand ist ein interner Zustand des Systems, der die Folge der Fehlerursache ist wie zum Beispiel ein falsch berechneter Zwischenwert.
- **Fehlerwirkung** (eng. failure)
Die Fehlerwirkung ist die nach außen hin sichtbare Auswirkung des Fehlzustandes, z.B. eine fehlerhafte Ausgabe am Bildschirm oder der Ausfall des Systems.

Nicht jeder Fehlzustand resultiert zwangsläufig in einer direkt verbundenen Fehlerwirkung. Fehlzustände treten nur auf, wenn eine Fehlerursache ausgelöst wird und können selbst dann weit entfernt von der eigentlichen Ursache zum Tragen kommen. Im Falle von

Mehrfachfehlern können diese sich gegenseitig beeinflussen wodurch die Identifizierung der Fehlerursache zusätzlich erschwert wird.

Fehlertolerante Systeme können Fehler erkennen und entsprechende Gegenmaßnahmen ergreifen. Fehler zu erkennen und eine detaillierte Diagnose über Ursache und potentielle Auswirkungen zu erstellen, ist Aufgabe der Fehlerdiagnose, welche im nächsten Abschnitt näher erläutert wird. Zunächst sollen jedoch die Möglichkeiten der Fehlerbehandlung geschildert werden.

Die Verfahren der Fehlerbehandlung können in drei Klassen eingeteilt werden [2]:

- **Fehlerbehebung**
Eine Fehlerbehebung entfernt den ursprünglichen Fehlzustand beispielsweise durch das Neustarten eines bestimmten Systemteils. Ein Beispiel hierfür ist das Recovery Scheme Block-Verfahren.
- **Rekonfiguration**
Bei der Rekonfiguration wird die fehlerhafte Komponente vom weiteren Systemablauf ausgeschlossen. So können zum Beispiel auszuführende Aufgaben auf noch intakte Komponenten verteilt werden (sogenannte Graceful Degradation). Darüber hinaus gibt es die Möglichkeit, die fehlerhafte Komponente durch eine Ersatzkomponente (Standby-Komponente) zu ersetzen.
- **Kompensation**
Bei der Fehlerkompensation wird der fehlerhafte Zustand nicht beseitigt. Stattdessen wird der Fehler von anderen Komponenten, einer höheren Ebene ausgeglichen. Ein klassisches Beispiel hierfür sind sogenannte N-aus-M Systeme mit Mehrheitsentscheid.

1.1 Fehlerdiagnose

Das Erstellen einer Fehlerdiagnose besteht aus zwei Schritten. Zunächst muss der Fehler erkannt werden, bevor seine Ursache weiter eingegrenzt werden kann.

Zur **Fehlererkennung** existieren verschiedene Möglichkeiten wie beispielsweise Replikationstests, Konsistenztests, Umkehrungstests, Prüfsummen oder Zeitüberprüfungen. Die Verfahren können in unterschiedlicher

*Diese Arbeit wurde durchgeführt bei der Firma IT-Designers GmbH, Esslingen

Kombination und Ausprägung zum Einsatz kommen [3].

Für eine Beurteilung des Systemzustandes wird häufig ein Modell des Systems verwendet. Das Modell kann beispielsweise auf Basis des zugrunde liegenden physikalischen Prozesses erstellt werden, in dessen Kontext das System betrieben wird. So kann erkannt werden, ob der Zustand eines Systems plausibel oder unplausibel ist und welcher Teil eines Systems den Fehler verursacht. Diese sogenannten **modellbasierten** Ansätze hängen jedoch stark vom verwendeten Modell ab. Das Modell muss zusätzlich zum System entwickelt werden und kann somit auch selbst fehlerhaft sein. Bei Änderungen am ursprünglichen System muss zudem auch das Modell angepasst werden. Nur ein detailliertes und möglichst fehlerfreies Modell eignet sich für eine präzise Fehlerdiagnose [4].

Der Systemzustand kann auch anhand einer längeren Beobachtung des Systems beurteilt werden. Auf Basis vergangener Zustandswerte kann der Plausibilitätsbereich für zukünftige Werte festgelegt werden. So kann ungewöhnliches Verhalten erkannt werden. In diesem Kontext werden unter anderem statistische Methoden, Trendanalysen und neuronale Netze verwendet. Diese Methoden arbeiten unabhängig von einem zugrunde liegenden spezifischen Systemmodell und können daher universeller entwickelt werden [4].

SFL bietet einen Ansatz innerhalb dieser sogenannten **beobachtungsbasierten** Fehlerdiagnose (engl. observation based). Hier wird die Beteiligung der Komponenten am Systembetrieb analysiert um dadurch fehlerhafte Komponenten zu lokalisieren. Wie das Verfahren funktioniert wird im Folgenden erläutert.

2 Spectrum-based Fault Localization

SFL analysiert die Beteiligung von Komponenten während der Systemausführung. Das Verfahren basiert auf der Annahme, dass eine Komponente die häufiger in Zusammenhang mit fehlerhaften Systemabläufen (**Transaktionen**) verwendet wird, wahrscheinlicher fehlerhaft ist, als Komponenten, die mit korrekten Ausführungen verwendet werden.

Der Anteil einzelner Komponenten an Transaktionen wird ausgewertet. Treten Fehler im System auf, kann eine Rangordnung erstellt werden, welche der Komponenten höchstwahrscheinlich den Fehler verursacht.

Für das Verfahren SFL werden folgende Parameter definiert:

- M Potentiell fehlerhafte Komponenten C
- N Transaktionen T
- N binäre Transaktionsergebnisse O
- Eine Aktivitätsmatrix A der Größe N x M

Für jede Transaktion kann ein Aktivitätsvektor gebildet werden, welche für jede Komponente die Information enthält, ob sie an der Transaktion beteiligt war oder nicht. Der letzte Vektor enthält die binäre Information, ob die jeweilige Transaktion erfolgreich war oder nicht. Aus den Aktivitätsvektoren entsteht eine Aktivitätsmatrix wie in folgender Abbildung dargestellt:

C	t ₁	t ₂	t ₃	t ₄	t ₅	t ₆
C ₀	1	1	1	1	1	1
C ₁	1	1	1	1	1	1
C ₂	1	1	1	1	0	1
C ₃	1	1	1	1	0	0
C ₄	1	1	1	1	0	1
C ₅	1	1	0	0	0	0
C ₆	1	1	1	1	0	1
C ₇	0	1	0	1	0	0
C ₈	1	0	1	0	0	1
C ₉	1	0	1	0	0	1
C ₁₀	1	1	1	1	1	1
O	1	1	1	1	0	0

Abbildung 1: Beispielhafte Aktivitätsmatrix

Anhand der Aktivitätsmatrix kann mithilfe binärer Vergleichsverfahren die fehlerhafte Komponente ermittelt werden. Dabei wird O mit dem Beteiligungsvektor (Spektrum) der Komponente an den verschiedenen Transaktionen verglichen. Je größer die Übereinstimmung desto wahrscheinlicher handelt es sich um die fehlerhafte Komponente [5].

Der geringe Bedarf an Ressourcen macht SFL zu einem effektiven Verfahren für die Fehlerlokalisierung zur Laufzeit. Anhand von binären Beteiligungsvektoren kann eine Aussage getroffen welche Komponente möglicherweise den Fehler verursacht. Das Verfahren benötigt dafür kein Modell des Systems und kann auch Mehrfachfehler lokalisieren. Somit kann mit geringem Implementierungsaufwand eine sehr exakte Fehlerdiagnose integriert werden [5].

Eine Herausforderung des Verfahrens bleibt der Overhead, der durch die Beobachtung des Systems entsteht. Chen et al. [6] verwenden SFL für Services mithilfe des Frameworks Turmeric. Das verwendete online Monitoring verursacht einen Overhead von 700% im überwachten System (Zielsystem) [6].

Das Entwickeln neuer Methoden zur effektiven Überwachung eines Systems ist Gegenstand aktueller Forschung und Entwicklung der ITDesigners GmbH und der Hochschule Esslingen. Im nächsten Kapitel wird ein Ansatz einer Arbeit beschrieben, wie er momentan entwickelt und evaluiert wird.

3 Dynamisches Monitoring für Publish/Subscribe-Systeme

In Publish/Subscribe-Systemen (P/S-Systeme) kommunizieren die verschiedenen

Komponenten eines Systems über ein gemeinsames Kommunikationssystem. Komponenten veröffentlichen (eng. publishen) Nachrichten auf diesem System unter bestimmten Kanälen (eng. Topics) und können gleichzeitig bestimmte Kanäle abonnieren (eng. subscriben). Veröffentlichten Komponenten Nachrichten unter einem Kanal, erhalten alle Abonnenten die Daten [7].

Publish/Subscribe-Systeme erlauben eine sehr entkoppelte Systemarchitektur. Dadurch ist es möglich ein System sehr dynamisch aufzubauen und beispielsweise bestimmte Teilkomponenten zur Laufzeit auszutauschen. Damit eignen sich P/S-Architekturen gut für dynamische und fehlertolerante Systeme.

Zudem ist es sehr einfach möglich das Kommunikationssystem zu überwachen. Dies ist essentiell für die Diagnose eines Systems. Überwachungskomponenten (Monitore) können als zusätzliche Interessenten auf den verschiedenen Kanälen mithören und die Daten auswerten. Die Auswertung der Überwachung kann somit offline stattfinden wodurch der Overhead im Zielsystem durch das Überwachungssystem deutlich reduziert werden kann.

In einer aktuellen Arbeit wird dieser Ansatz untersucht. Im Rahmen der Arbeit wird ein Framework entwickelt, um P/S-Systeme zu überwachen. Für die Fehlerdiagnose mit SFL ist es notwendig Abläufe in einem System zu verfolgen (**Tracing**). Transaktionen können je-

doch nicht nur streng sequentiell sondern auch parallel und asynchron verlaufen. Während sequentielle Abläufe eines System relativ einfach zu verfolgen sind, gestaltet sich das Verfolgen von parallelen Abläufen weitaus schwieriger. Zudem ist in einem P/S-System die Überwachung auf das Kommunikationssystem beschränkt. Werden innerhalb einer Komponente des Systems Daten asynchron verarbeitet, können Transaktionen nicht so einfach verfolgt werden.

Für die verschiedenen Arten von Transaktionen gibt es verschiedene Methoden um sie zu verfolgen. Diese unterscheiden sich am Grad des Systemeingriffs. Die verschiedenen Ansätze unterteilt Deen [8] in drei Klassen, die durch das Framework realisiert werden können:

- **Monitoring ohne Systemeingriff (eng. non-intrusive):** Monitore überwachen lediglich das Kommunikationssystem.
- **Monitoring mit leichtem Systemeingriff (eng. weak-intrusive):** Der Nachrichtenverkehr des Zielsystems wird zur Überwachung manipuliert.
- **Monitoring mit Systemeingriff (eng. intrusive):** Das Zielsystem wird weitergehend verändert, beispielsweise werden die Komponenten um einen ID-Mechanismus ergänzt.

-
- [1] J.-C. Laprie, Ed, Dependability: basic concepts and terminology, Wien: Springer-Verlag, 1992.
 [2] W. Görke and A. Endres, Fehlertolerante Rechensysteme. München: Oldenbourg, 1989.
 [3] W. Torrespomales, Software Fault Tolerance: A Tutorial, pp. 2000-210616
 [4] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, A review of process fault detection and diagnosis, Computers and Chemical Engineering, vol. 27, no. 3, pp. 293-311, 2003.
 [5] R. Abreu, P. Zoetewij, and A. J. C. van Gemund, An observation-based model for fault localization, in the 2008 international workshop on dynamic analysis, p. 64.
 [6] C. Chen, H.-G. Gross, and A. Zaidman, Improving Service Diagnosis through Increased Monitoring Granularity, in 2013 7th IEEE International Conference on Software Security and Reliability (SRE), pp. 129-138.
 [7] Vargas, Pesonen et al. 2007 - Transactions in Content-Based Publish Subscribe Middleware
 [8] Gert-Jan Deen 2012 - Dynamic Analysis of SOA, Master-Thesis, TU Delft
 [9] <http://www.redis.io/>

Konzeption und Realisierung der Erweiterung einer Seriensoftware um eine Schnittstelle zu CAN FD für ein als Gateway konzipiertes Automotive-Steuergerät einschließlich der Entwicklung des Hardware Abstraction Layers

Christian Eissler*, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In der heutigen Zeit entstehen Innovationen im Automobil hauptsächlich in der Elektronik und Software. Moderne Fahrzeuge integrieren oft bis zu 70 und mehr Steuergeräte. Meist ist die Realisierung von Steuergeräten während der Fahrzeugentwicklung sehr zeit- und kostenintensiv. Deswegen wird zu Beginn der Entwicklung oft kostengünstig ein Prototyp erstellt, um Anforderungen schnell umsetzen zu können oder bei Kleinserien komplett auf die Neuentwicklung eines Steuergerätes zu verzichten.

Mit der GIGABOX hat GIGATRONIK ein flexibel einsetzbares prototypisches Steuergerät entwickelt. In der Fahrzeugentwicklung kann dieses zur preisgünstigen Realisierung von Steuergeräte-Applikationen eingesetzt werden und Anforderungen lassen sich einfach und schnell umsetzen. Die GIGABOX besteht aus einem Grundmodul und kann durch eine kundenspezifische Applikationsplatine erweitert werden. Das Grundmodul verfügt über einen leistungsfähigen Prozessor und eine Vielzahl von Standardschnittstellen, die im Automobilbereich Verwendung finden. Programmiert werden kann die GIGABOX über eine einfach anzuwendende Skriptsprache, die nur grundlegende Programmierfähigkeiten voraussetzt und eine Programmierung auch ohne umfangreiche Kenntnisse der Hardware ermöglicht.

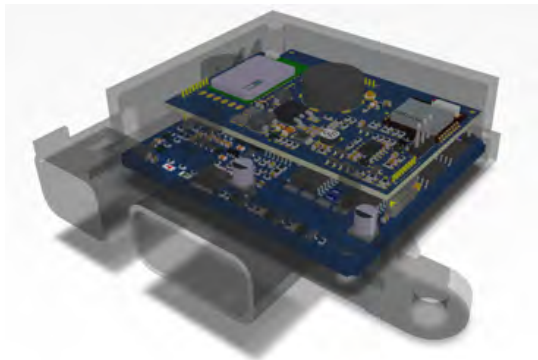


Abbildung 1: GIGABOX FD

Die Vielzahl der Steuergeräte und das weiter wachsende Datenaufkommen bringen herkömmliche Bussysteme wie z.B. CAN an ihre Grenzen. Um dem entgegenzuwirken wurde der Standard CAN FD entwickelt. Bei diesem handelt es sich um eine Protokollerweiterung von CAN, die eine größere Nutzdatenlänge, sowie eine höhere Übertragungsrate ermöglicht. Dadurch ist ein bis zu 8-fach höherer Datendurchsatz möglich. Seit der Spezifikation von CAN FD im Jahr 2012 finden immer mehr Steuergeräte mit CAN FD Anwendung im Fahrzeug [1].



Abbildung 2: CAN FD

Im Rahmen dieser Bachelorarbeit wurde die Software der neuen GIGABOX FD konzipiert und entwickelt. Hierzu musste ein Hardware Abstraction Layer entwickelt werden, um die Peripherie des Mikrocontrollers anzubinden, sowie die Applikation und der Bootloader der bestehenden GIGABOX angepasst werden. Die neue GIGABOX FD wurde um eine Schnittstelle zu CAN FD erweitert, soll ältere Produktvarianten ablösen, neue Technologien integrieren, eine höhere Performance bieten und innovative Produktkonzepte umsetzen.

Für die GIGABOX FD gibt es eine Vielzahl an Einsatzgebieten. In der Fahrzeugentwicklung wird zum Beispiel oft als Basis für eine Neuentwicklung eine alte Netzwerktopologie (altes Fahrzeug) eingesetzt. Die neu entwickelte Hardware ermöglicht aber oft keinen direkten Zugriff auf die alte Netzwerktopologie aufgrund von inkompatiblen Technologien (Schnittstellen). In einem solchen Fall kann die GIGABOX FD als Gateway eingesetzt werden, um einen Austausch von Nachrichten zwischen diesen Schnittstellen zu ermöglichen. Abbil-

*Diese Arbeit wurde durchgeführt bei der Firma GIGATRONIK Stuttgart GmbH, Stuttgart

dung 3 zeigt als Beispiel die Einsatzmöglichkeit der GIGABOX FD als Gateway zwischen CAN und CAN FD.

Die GIGABOX FD bietet neben CAN und CAN FD auch analoge Eingänge, digitale Ein- und Ausgänge, Leistungsschalter und eine LIN-Schnittstelle. Durch diese umfangreiche Konnektivität kann mit der GIGABOX FD auch die komplette Steuergerätefunktionalität nachgebildet werden, während das endgültige Steuergerät noch nicht verfügbar ist. Durch die einfache Programmierbarkeit ist die GIGABOX FD optimal für das Rapid Prototyping geeignet.

Als Entwicklungsumgebung für die GIGABOX wird das von GIGATRONIK entwickelte Softwaretool `configurAIDER` eingesetzt. Die-

ses stellt einen Editor und einen Compiler für die Skriptsprache bereit, sowie eine alternative Möglichkeit zur Konfiguration der GIGABOX mittels WENN-DANN-SONST Anweisungen. Als Skriptsprache wird PAWN eingesetzt, welche auf einer C-ähnlichen Syntax basiert. Diese erzeugt einen Bytecode, der auf jeder GIGABOX lauffähig ist.

Die GIGABOX FD kann über die USB-HID-Schnittstelle (USB Human Interface Device Schnittstelle) konfiguriert und programmiert werden. Vorteil dieser Schnittstelle ist, dass keine zusätzlichen Gerätetreiber installiert werden müssen und die GIGABOX FD so auf jedem PC mit USB-Schnittstelle sofort verwendet werden kann.

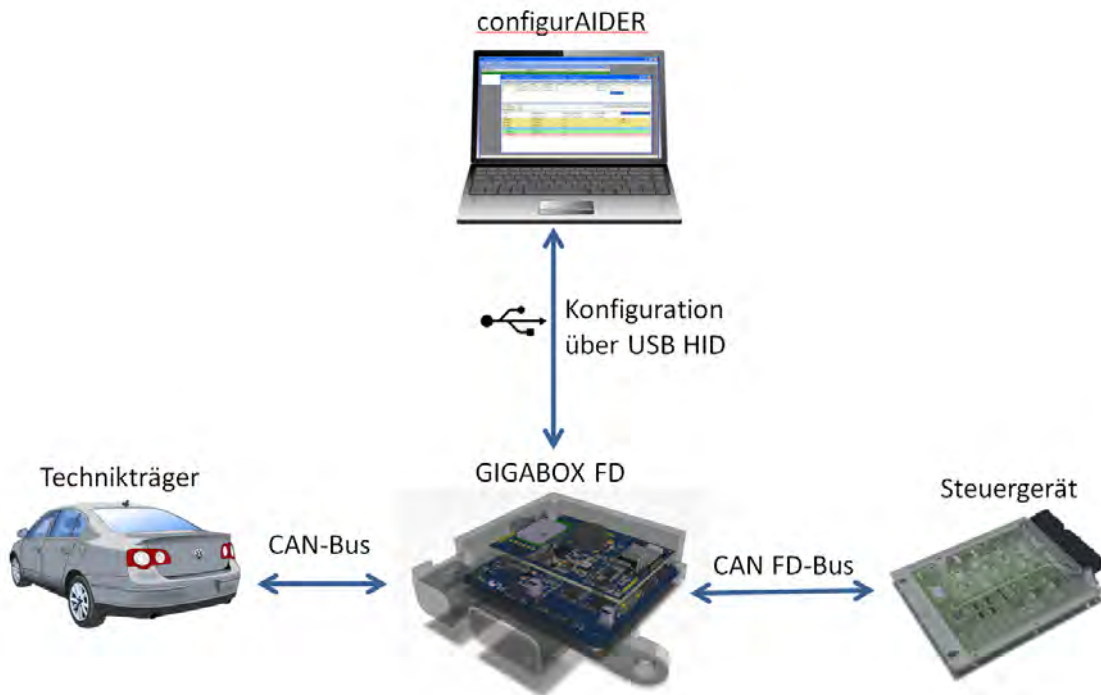


Abbildung 3: Einsatzmöglichkeit der GIGABOX FD als Gateway

[1] Robert Bosch GmbH: Spezifikation CAN FD Version 1.0, http://www.bosch-semiconductors.de/media/ubk_semiconductors/pdf_1/canliteratur/can_fd_spec.pdf

Bildquellen:

- Abbildung 1: GIGATRONIK Stuttgart GmbH
- Abbildung 2: Robert BOSCH GmbH [1]
- Abbildung 3: Christian Eissler (Grafiken: GIGATRONIK Stuttgart GmbH, openclipart.org)

Entwurf und prototypische Realisierung der Software unter FreeRTOS zu einem Test-Interface für das AC-System von Festo

Daniel Fröhlich*, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Aufgabe der vorliegenden Bachelor-Abschlussarbeit ist der Entwurf und die Realisierung einer Software, die die Speicherregister des AC-Systems von Festo visualisiert. Das Ziel des Projekts ist zum einen eine übersichtliche Gliederung der einzelnen Register und zum anderen die Realisierung einer Anbindung für Python-Skripts, so dass Tests, die in der Script-Sprache Python geschrieben werden, die Funktionen einzelner Peripherie-Bausteine des AC-Systems verifizieren können. Dabei tauscht die Software die relevanten Daten und Speicheradressen über eine USB-Schnittstelle mit einem Mikrocontroller der Familie FM4 von Spansion aus. Dieser wiederum stellt eine Verbindung über eine QuadSPI-Schnittstelle zu einem FPGA her, auf dem das AC-System realisiert ist.

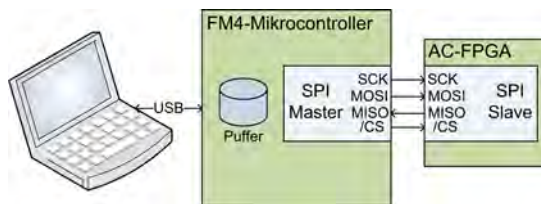


Abbildung 1: Aufbau des Projekts

Bei dem AC-System handelt es sich um ein von Festo entwickeltes Kommunikationssystem, das an überlagerte Feldbusse bzw. Industrial Ethernet gekoppelt werden kann. Dabei ist dieses System echtzeitfähig und arbeitet nach dem Master-Slave-Prinzip, d.h. es gibt einen Kommunikationsmaster, der die Kommunikation steuert.

Das AC-System verfügt über zwei Arten von Registern: System-Register und Applikationsregister. Der wesentliche Unterschied zwischen beiden ist die Tatsache, dass von außerhalb nicht auf die System-Register zugegriffen werden kann. Daher werden im vorliegenden Projekt nur die Applikationsregister dargestellt, wie beispielsweise LED, GPIO, usw.

Das Herzstück des AC-Systems ist ein ASIC, das in jedem Modul eingebaut wird und dort die Kommunikation steuert. Ein ASIC (Application Specific Integrated Circuit) ist ein Baustein,

mit dem sich komplexe und aufwändige Schaltungen auf nur einem Chip integrieren lassen.

Bei der vorliegenden Abschlussarbeit wurde anstelle des ASICs jedoch noch ein FPGA genutzt. Auf einem FPGA (Field Programmable Gate Array) lassen sich diese logischen Schaltungen programmierbar realisieren, es ist gewissermaßen ein „SoftASIC“ mit großen Vorteilen während der Entwicklung der Funktionen des ASICs.

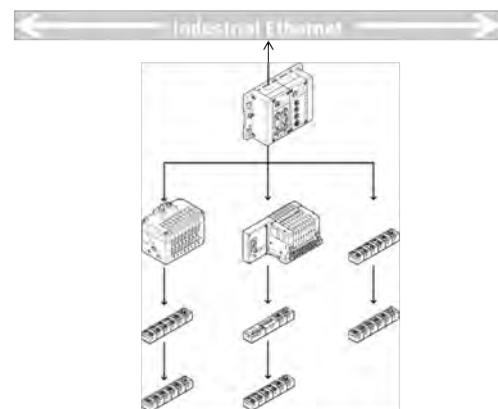


Abbildung 2: AC-System in Verbindung mit diversen Modulen und überlagertem Industrial Ethernet

Die Konfiguration des FPGAs erfolgt im Wesentlichen durch die Aktivierung und Deaktivierung von Verbindungen zwischen einzelnen Logikelementen und Logikblöcken. Sie erfolgt entweder grafisch anhand eines Schaltplans oder durch eine Beschreibung der Strukturen und Abläufe einer Schaltung mit Hilfe einer Hardwarebeschreibungssprache, wie beispielsweise VHDL oder Verilog.

Das für die Verbindung mit dem ASIC eingesetzte Serial Peripheral Interface (SPI) ist ein einfaches auf dem Master-Slave-Prinzip aufbauendes Bus-System, das für eine synchrone serielle Kommunikation genutzt wird. Hierbei sind die Kommunikationsteilnehmer über vier gemeinsame Leitungen miteinander verbunden:

- SCK (Serial Clock): Hierüber steuert der Master die Taktfrequenz und damit die Synchronisation

*Diese Arbeit wurde durchgeführt bei der Firma Festo AG & Co. KG, Sirnauer Brücke

- MISO (Master Input, Slave Output): Über diese Leitung werden Daten vom Slave zum Master gesendet
- MOSI (Master Output, Slave Input): Diese Leitung überträgt die Daten vom Master zum Slave
- CS (Chip Select): Auf dieser Leitung wird der jeweilige Slave ausgewählt, mit dem die Daten ausgetauscht werden sollen. Es handelt sich hierbei um eine Leitung mit negativer Logik, d.h. die Slaves werden nicht mit einer binären Eins, wie bei den oben erwähnten Leitungen, sondern einer binären Null ausgewählt.

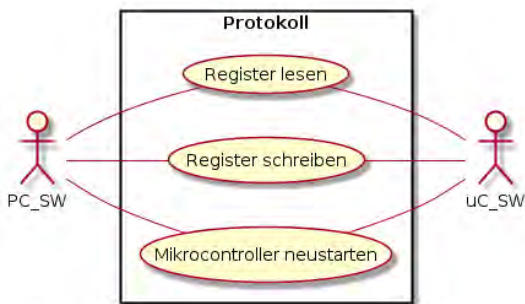


Abbildung 3: Use Case-Diagramm des Übertragungsprotokolls

Im vorliegenden Projekt agiert der Mikrocontroller als SPI-Master und das AC-FPGA als SPI-Slave (siehe Abbildung 1).

Bei dem im Projekt eingesetzten Mikrocontroller handelt es sich um einen ARM Cortex-M4 mit einem 32-Bit-Prozessorkern. Die ARM Cortex-M4-Architektur baut auf der des ARMv7-Chips auf, welcher sehr häufig in mobilen Geräten, wie Smartphones und Tablets, eingesetzt wird. Hier wird ein Speicher reserviert, um die Register-Adressen sowie deren Werte in einem Ringpuffer zwischenspeichern. Des Weiteren sorgt der Puffer dafür, dass mehrere Daten gleichzeitig angefordert werden können.

Der USB (Universal Serial Bus) ist ebenso ein serielles Bussystem. Es wird in der Informationstechnik für die Verbindung von Computern mit externen Geräten, wie etwa Tastatur, Maus oder Speichermedien, genutzt. Der USB kann, verglichen mit anderen seriellen Bussystemen, sehr hohe Datentransferraten erreichen, bei der aktuellsten Version des USB, USB 3.0 SuperSpeed+, bis zu 10 GBit/s.

Der USB-Standard definiert zwei Geräteklassen: das Gerät und das Interface. Der Mikrocontroller dieser Abschlussarbeit beispielsweise agiert als Interface und ist in der so genannten „Communications Device Class“ (CDC) einzuordnen.

Zur sicheren Datenübertragung über die USB-Schnittstelle des Mikrocontrollers wird im Rahmen dieser Abschlussarbeit ein speziell auf dieses Verfahren angepasstes Übertragungsprotokoll implementiert. Dieses soll relativ klein und einfach gehalten werden, um unnötigen Overhead zu vermeiden und eine höhere Geschwindigkeit zu gewährleisten.



Abbildung 4: Logo des FreeRTOS-Betriebssystems

Die komplette Software wird auf dem speziell für eingebettete Systeme angepassten Betriebssystem FreeRTOS ausgeführt. Der Name RTOS steht für „Real Time Operating System“, woraus sich auch direkt ableiten lässt, dass es sich um ein Echtzeitbetriebssystem handelt. Durch den Namenszusatz „Free“ (engl. für frei) wird signalisiert, dass es sich um ein Open-Source-Projekt handelt.

- [1] Festo AG & Co. KG (Hrsg.): Protokoll-Spezifikation
- [2] <http://www.halbleiter-scout.de/informationen/elektronik/346-asic.html>
- [3] <http://www.aufzu.de/FPGA/kochbuch/kap2a.html>
- [4] <http://www.mct.de/faq/spi.html>
- [5] <http://www.arm.com/products/processors/cortex-m/>
- [6] <http://www.elektronik-kompodium.de/sites/com/0312021.htm>
- [7] <http://www.freertos.org/about-RTOS.html>

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Festo AG & Co. KG (Hrsg.): Protokoll-Spezifikation
- Abbildung 3: Eigene Darstellung
- Abbildung 4: <http://www.freertos.org/logo.jpg>

Vergleich der Protokolle AKEP2 und ECDH für fahrzeuginternen Peer-to-Peer Schlüsselaustausch

Jan Hammer*, Reinhard Schmidt, Dominik Schoop

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Durch die zunehmende Vernetzung innerhalb und um das Fahrzeug, spielt der Bereich Automotive Security eine immer bedeutendere Rolle. Um durch den Einfluss von außen, wie z.B. bei Car2x, die interne Kommunikation nicht zu gefährden, hat der AUTOSAR Standard in Version 4.2.1 das Modul Secure On Board Communication, kurz SecOC, eingeführt [1]. Mit diesem Modul ist es möglich, Nachrichten innerhalb der Fahrzeugnetzwerke per Authentifizierung abzusichern. Üblicherweise wird dies über Message Authentication Codes (MAC) erreicht. Für dieses kryptographische Verfahren werden Schlüssel benötigt, die wiederum verwaltet werden müssen. Dabei ist es nicht nur relevant, dass diese Schlüssel initial in die Steuergeräte programmiert werden, was als Initial Keying bezeichnet wird, sondern zukünftig auch ausgetauscht werden können, was als Re-Keying bezeichnet wird. Beispielsweise kann bei einem Werkstatt-Service die Notwendigkeit neue Schlüssel einzuspeisen bestehen, z.B. wenn ein defektes Steuergerät getauscht wird. Ein weiterer Grund für Re-Keying ist die Erhöhung der Sicherheit, durch kürzere Kryptoperioden der Schlüssel.

Es gibt verschiedenste Möglichkeiten Keying durchzuführen, wobei die Protokolle für den Schlüsselaustausch grundsätzlich in Peer-to-Peer und Master-Slave Verfahren eingeteilt werden können. In dieser Arbeit beschränkt sich die Analyse auf den Peer-to-Peer Austausch, bei dem jeder Teilnehmer einen gleichberechtigten Anteil am Vorgang hat. Das Verfahren soll anhand von zwei Protokollen betrachtet werden. Eines auf Basis symmetrischer Kryptographie und eines auf Basis von asymmetrischer Kryptographie. Das betrachtete asymmetrische Verfahren ist der Schlüsselaustausch per Elliptic Curve Diffie Hellman (ECDH). Dem wird das Authenticated Key Exchange Protocol 2 (AKEP 2) gegenüber gestellt, das beim Schlüsselaustausch rein auf symmetrische Kryptographie setzt.

Voraussetzung für die Teilnehmer ist bei beiden Protokollen ein gemeinsames Geheimnis. Dieses muss im Voraus bei den Kommunikationsteilnehmern eingespeist werden. Im Falle von ECDH handelt es sich hierbei um ein Zerti-

ifikat, mit dem die Gegenseite den Teilnehmer eindeutig identifizieren und als vertrauenswürdig einstufen kann. Bei AKEP 2 werden für jeden Teilnehmer zwei symmetrische Schlüssel eingespeichert. Einer davon dient ausschließlich zur Authentifizierung, mit dem Zweiten wird unter Zunahme der ausgetauschten Parameter der Session Key erzeugt.

Beide Verfahren benötigen drei Nachrichten um einen beidseitigen Austausch durchzuführen. Diese lassen sich für beide Protokolle folgendermaßen verallgemeinern:

1. $A \Rightarrow B$: Anfrage für den Austausch mit Zufallszahl.
2. $A \Leftarrow B$: Antwort mit Parametern zur Schlüsselberechnung, sowie der erhaltenen Zufallszahl und Authentifizierung der Daten.
3. $A \Rightarrow B$: Authentifizierung von A, um B die Identität zu bestätigen.

Bei AKEP2 werden die Schlüssel über Zufallszahlen berechnet. Es werden somit mehrere zufällige Zahlenwerte während des Protokollablaufs ausgetauscht. Der MAC zur Authentifizierung wird über die Daten berechnet. Bei ECDH wird mit den übertragenen Parametern sowohl die elliptische Kurve als auch das Verfahren mit dem der öffentliche und der geheime Schlüssel generiert werden, festgelegt. Die Authentifizierung wird durch eine digitale Signatur umgesetzt. Hierfür wird das eingespeicherte Zertifikat verwendet.

Aktuelle Kryptobibliotheken enthalten Implementierungen von Diffie-Hellman mittels elliptischer Kurven (ECDH). Allerdings ist das noch nicht lange so, weshalb ECC-Verfahren noch nicht weit verbreitet sind [2]. Besonders interessant sind ECC-Verfahren, weil beim Diffie Hellman Verfahren in Verbindung mit elliptischen Kurven dieselbe Stufe an Sicherheit mit wesentlich kleineren Schlüsseln erreicht werden kann als bei vergleichbaren Verfahren. Dadurch kann Speicher und Bandbreite bis zu einem Faktor 30 eingespart werden, siehe dazu Abbildung 1 [3]. Dargestellt wird, wie groß der öffentliche Schlüssel gewählt werden muss, um eine zu symmetrischen Schlüsseln vergleichbare Sicherheit zu erhalten. In die-

*Diese Arbeit wurde durchgeführt bei der Firma Vector Informatik GmbH, Stuttgart

sem Beispiel sind die Angaben auf die symmetrische Blockchiffre Advanced Encryption Standard (AES) bezogen. Die Länge des hierfür verwendeten Schlüssels ist ein Indikator für die Sicherheit des Verfahrens.

ECC KEY SIZE (Bits)	RSA KEY SIZE (Bits)	KEY SIZE RATIO	AES KEY SIZE (Bits)
163	1024	1 : 6	
256	3072	1 : 12	128
384	7680	1 : 20	192
512	15 360	1 : 30	256

Supplied by NIST to ANSSI X9F1

Abbildung 1: NIST guidelines for public key sizes for AES

Sowohl ECDH als auch AKEP2 sind so entworfen, dass diese Protokolle vor Replay-Attacken und Man-in-the-Middle Angriffen geschützt sind [4], vgl. Abbildung 2.

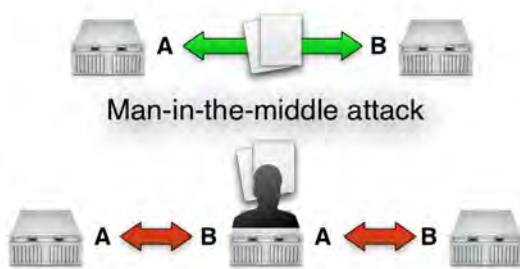


Abbildung 2: Man in the Middle Attacke

Bei Replay-Attacken wird eine frühere Sitzung aufgezeichnet und Nachrichten daraus an einen der Kommunikationsteilnehmer gesendet. Dadurch soll eine Anfrage eines anderen Teilnehmers vorgetäuscht werden mit

dem Ziel, unterschiedliche Session-Keys in beiden Kommunikationsteilnehmern zu erreichen. Dadurch ist die abgesicherte Kommunikation zwischen den Teilnehmern im Endeffekt zerstört. Ein Man-in-the-Middle Angriff liegt vor wenn eine dritte Partei sich zwischen die Kommunikation zweier Teilnehmer schaltet. Währenddessen gibt der Angreifer für jeden Teilnehmer vor, die Identität der Gegenseite zu haben. Dadurch kann der Datenverkehr mitgelesen und manipuliert werden. Die Protokolle bieten außerdem Perfect Forward Secrecy, wodurch bei einem Verlust eines Sitzungsschlüssels sowohl vorhergehende, als auch nachfolgende Sitzungen nicht kompromittiert sind [5]. Ziel der Arbeit ist es unter anderem diese beiden Protokolle auf ihre Umsetzbarkeit im Automotive Umfeld zu untersuchen. Dazu wird auf Basis des Vector Tools CANoe ein Demonstrator genutzt, mit dem es möglich ist, Steuergeräte und Netzwerke zu simulieren und zu überwachen. Darüber hinaus ist ein Steuergerät mit Hilfe einer Embedded Linux Hardware umgesetzt worden, auf Basis eines BeagleBoards. Hiermit lassen sich belastbare Zeitmessungen durchführen um die Umsetzbarkeit der Protokolle zu ermitteln. Außerdem wird betrachtet, wie sich beide Protokolle bezüglich der benötigten Buslast unterscheiden, also wie viele Daten zwischen den Steuergeräten gesendet werden müssen. Abgerundet wird der Vergleich beider Protokolle bezüglich der Fähigkeit, wie diese hinsichtlich mehrerer Steuergeräte skalieren. Das ist notwendig, da bei den üblichen Client-Server Anwendungen im Internet andere Anforderung an die Anwendungen gestellt werden, als in einem Steuergerätenetzwerk im Fahrzeug, in dem eine Vielzahl von Teilnehmern gleichzeitig miteinander kommunizieren.

- [1] AUTOSAR_TR_ReleaseOverviewAndRevHistory: <http://www.autosar.org/fileadmin/files/releases/4-2>
 [2] <http://www.elektronik-kompodium.de/sites/net/1910151.htm>
 [3] <https://www.certicom.com/index.php/the-basics-of-ecc>
 [4] <http://www.elektronik-kompodium.de/sites/net/1710251.htm>
 [5] <http://www.elektronik-kompodium.de/sites/net/1809181.htm>

Bildquellen:

- Abbildung 1: <https://www.certicom.com/index.php/the-basics-of-ecc>
- Abbildung 2: <http://www.4kcc.com/images/maninthemiddleattack.png>

Ermittlung der Fahrzeuglage mittels Sensordatenfusion

Tobias Michael Heine*, Hermann Kull

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die zunehmende Funktionalität von modernen Fahrzeugen erfordert die Erfassung der Umwelt von Fahrzeugen mittels Positions- und Inertialsensorik. So spielt auch die Bestimmung der Lage des Fahrzeuges für zukünftige Funktionalitäten eine zentrale Rolle. Im Rahmen der Bachelor-Abschlussarbeit

soll die Lage des Fahrzeuges mit Hilfe der im Versuchsträger verbauten Inertialsensorik bestimmt werden. Abbildung 1 veranschaulicht die Achsen sowie die zugehörigen Winkel mit deren entsprechenden Bezeichnungen am Fahrzeug.



Abbildung 1: Bezeichnungen der Achsen am Fahrzeug

Inertialsensorik

Der eingesetzte Inertialsensor SMI130 von Bosch besteht aus einem Gyroskop und einem Beschleunigungssensor. Im Folgenden werden die Komponenten sowie deren Probleme bei der Sensorik erläutert.

Beschleunigungssensor

Mithilfe des Beschleunigungssensors lassen sich auftretende Beschleunigungen in [g] bzw. [m/s^2] messen. Die Berechnung des Roll- und Nickwinkels erfolgt über die Darstellung der gemessenen Beschleunigungsvektoren als Kräftevektoren der auf der jeweiligen Achse verbleibenden Erdanziehungskraft. Da die Erdanziehungskraft immer senkrecht zur Erdoberfläche steht, können mithilfe von trigo-

nometrischen Funktionen Roll- und Nickwinkel absolut zur Erdoberfläche aus den Kräftevektoren berechnet werden. Somit lassen sich Roll und Nickwinkel nach folgenden Gleichungen ermitteln:

$$\alpha_{\text{Roll}} = \text{atan2}\left(\frac{F_x}{\sqrt{F_z^2 + F_y^2}}\right) \cdot \frac{180}{P_i} \quad (\text{Gl.1})$$

$$\alpha_{\text{Nick}} = \text{atan2}\left(\frac{F_y}{\sqrt{F_z^2 + F_x^2}}\right) \cdot \frac{180}{P_i} \quad (\text{Gl.1})$$

Eine Bestimmung des Gierwinkels ist nicht möglich, da die Kräfte in Longitudinal und Lateralrichtung in derselben Ebene liegen. Die Winkelberechnung auf Grundlage der mit dem Beschleunigungssensor ermittelten Werte erweist sich sehr Anfällig für Vibrationen, Schläge sowie bei Bewegungen auftretende Kräfte.

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Leonberg

Gyroskop

Mit Hilfe von Gyroskopen lassen sich Winkelgeschwindigkeiten in $[\text{°}/\text{s}]$ messen. Die Bestimmung des relativen Winkels zur Ausgangslage erfolgt durch Integration der aktuellen Winkelgeschwindigkeit. Als Problematisch bei der Bestimmung des relativen Winkels zur Ausgangslage erweist sich die Gyroskop-Drift. Diese entsteht durch die Integration der Winkelgeschwindigkeit die einen Bias in Form eines Offset beinhaltet. Der Bias des Gyroskops ist nicht konstant und abhängig vom Luftdruck und der Temperatur. Durch die Fehlerfortpflanzung bei der Integration entsteht so eine Drift des integrierten relativen Winkels. So kann der Wert des relativen Winkels um mehrere 100° pro Stunde vom tatsächlichen Wert abweichen [2].

Sensordatenfusion

Da sowohl Beschleunigungssensor als auch Gyroskop die jeweils genannten Probleme bei der Sensorik aufweisen, sollen im Rahmen der Bachelor-Abschlussarbeit die Daten beider Sensoren mittels Sensordatenfusion fusioniert werden, um Roll- und Nickwinkel exakt zu bestimmen. Um die Daten von Beschleunigungssensor und Gyroskop zu fusionieren wird ein Kalman-Filter verwendet. Kalman Fil-

ter eignen sich ideal um aus verrauschten und teilweise redundanten Messungen, Zustände und Parameter eines Systems zu schätzen [1]. Im Rahmen der Studienarbeit wird ein Komplementär-Kalman-Filter eingesetzt. Dieses fusioniert die Daten aus Gyroskop und Beschleunigungssensor nach Gleichung 3, so dass eine präzise Bestimmung des absoluten Winkels ermöglicht wird.

$$\Theta_k = \Theta_{k-1} + \Delta t (\dot{\Theta}_{k-1} - \dot{\Theta}_{\text{Bias}_{k-1}}) \quad (Gl.3)$$

Nach Gleichung 1 ergibt sich der neue absolute Winkel aus dem letzten Winkel auf Basis der Beschleunigungswerte zuzüglich der relativen Winkeländerung. Aus Integration der Bias kompensierten Winkelgeschwindigkeit die mit dem Gyroskop erfasst wurde, ergibt sich die relative Winkeländerung des Sensors relativ zur Ausgangsposition. Da das Kalman-Filter auf Grundlage von Stochastischen Schätzverfahren arbeitet, ist es möglich, das verrauschte und un stabile Signal des Beschleunigungssensors zu plausibilisieren. Zudem wird durch die Schätzung des Gyroskop-Bias die bei der Integration der Gyroskop-Winkelgeschwindigkeit entstehende Gyroskop-Drift eliminiert. Somit entsteht mithilfe des komplementären Kalman Filters wie in Abbildung 2 dargestellt, eine stabile Schätzung des Roll- und Nickwinkels.

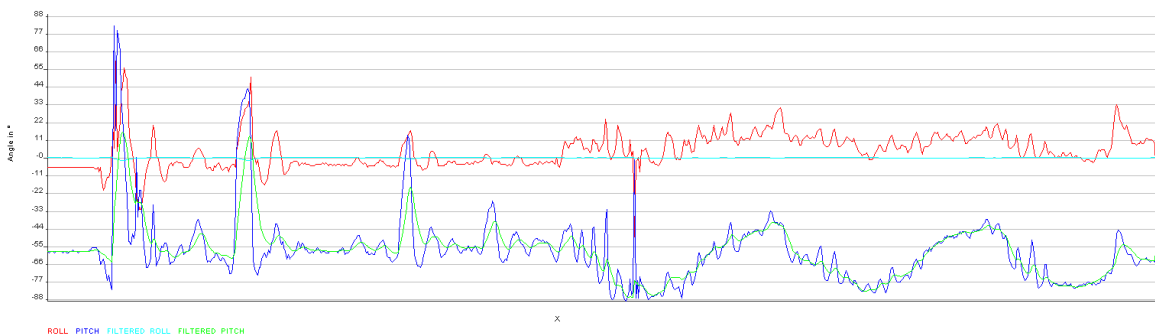


Abbildung 2: Mittels Kalman-Filter gefiltertes Signal (grün) des Nickwinkels (blau)

[1] Prof. Reiner Marchthaler, HS-Esslingen, Skript: Embedded Systems Software – Abschnitt Kalman-Filter

[2] LXRobotics GmbH, Kalmanfilter für die Sensorfusion von Gyroskop und Magnetometer,

URL: <https://www.lxrobotics.com/kalmanfilter-fuer-die-sensorfusion-von-gyroskop-und-magnetometer/>;
Zugriff: 10.11.2015

Bildquellen:

- Abbildung 1: Kollage mit images.thecarconnection.com
- Abbildung 2: Eigene Herstellung

Entwicklung von Komfort- und Servicefunktionen für ein „Neighborhood Electric Vehicle“

Daniel Herbst*, Harald Melcher, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Fahrzeugklasse der Leichtelektromobile LEM (englisch Neighborhood Electric Vehicle – NEV) ist bereits seit Jahrzehnten existent. Heutzutage gibt es rund eine halbe Million dieser NEV weltweit und jährlich kommen etwa weitere 50 Tausend hinzu [1]. Die meisten Fahrzeuge dieser Klasse verfügen über eine einfache und funktionelle Ausstattung. Die Höchstgeschwindigkeit liegt bei etwa 35 km/h. Die Weiterentwicklung dieser Fahrzeuge ist im Wesentlichen durch die Verwendung von edleren Materialien, wie feines Leder, Carbon und Chrom etc. geprägt und zielt damit hauptsächlich auf das Design der Fahrzeuge ab. Im Bereich der Elektrik/Elektronik Entwicklung hat sich bisher sehr wenig getan. Die letzte Innovation eines der Marktführer in den vergangenen Jahren war die Integration einer USB-Schnittstelle zur Ladung von mobilen Endgeräten. Ein 2013 durchgeführter internationaler Ideenwettbewerb zeigt, dass auch ein Bedarf an technischen Innovationen besteht [2]. In Abb. 1 ist das Mercedes-Benz Vision Golf Cart zu sehen, welches auf Grundlage dieser Ideen und Konzepte und unter Berücksichtigung einer Mercedes-Benz Designstrategie erstellt wurde.



Abbildung 1: Mercedes-Benz Vision Golf Car

Demnach birgt diese Fahrzeugklasse ein unheimlich großes Potential in sich. Es gibt Bedarf an Komfortfunktionen aus dem E/E-Bereich, wie ein ESP als Kippschutz für Fahrten im Gelände. Neben dem am meisten verbreiteten Ansatz im Golfsport und Tourismus gibt es noch andere Anwendungsfelder, wie beispielsweise der Einsatz auf autofreien Uni-

Campus, in Wohnkommunen – sogenannten Gated Communities – oder Städten, die neben Radwegen auch Wege für NEV's vorsehen [3].

Die als Nachrüstlösung erdachten Funktionen sollen in bestehende serienmäßige NEV's leicht nachrüstbar sein. Sie werden in zwei Kategorien eingeteilt, Sicherheits- und Komfortfunktionen. Als Sicherheitsfunktionen werden Eigenschaften bezeichnet, die zur Unfallvermeidung, Insassenschutz oder Diebstahlschutz verwendet werden. Darunter versteht man ESP, Airbag oder auch die Wegfahrsperrung. Komfortfunktionen sind Systeme, die dem Fahrer das Leben während der Fahrt, erleichtern sollen. Zu den komplexeren Systemen gehört der Einparkassistent, der entweder vollkommen autonom einparkt oder beim Einparken unterstützt. Die Sitzheizung ist dagegen eine eher einfache Variante, einer Komfortfunktion, die das Autofahren erleichtert. Da die Golf Carts vermehrt im Gelände auf Golfplätzen unterwegs sind und es dort Hügel und Senken gibt, kann es bei zu steilen Auffahrten oder Gefälle zu gefährlichen Situationen kommen bei dem das NEV umkippen kann. Dieser maximale Neigungswinkel ist dem Fahrer meist nicht bekannt und soll eine Lösung zur Nachrüstung sein. Es werden dabei die Sensordaten eines Beschleunigungssensor sowie eines Gyroskops verwendet, um die Neigung zu bestimmen und somit das Kippen des Fahrzeuges zu verhindern.

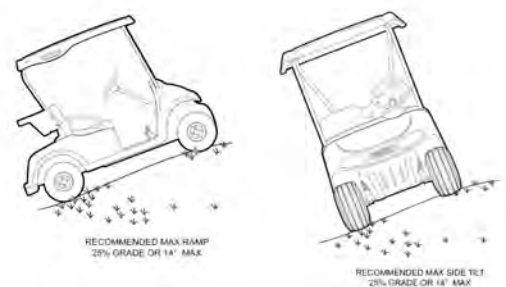


Abbildung 2: NEV an Schräglage

Als Nachrüstlösung wird bei den Komfortfunktionen eine Übertragung wichtiger Fahrzeuginformationen per Bluetooth Low Energie (BLE) angedacht. Dabei sollen nicht nur In-

*Diese Arbeit wurde durchgeführt bei der Firma GIGATRONIK Stuttgart GmbH, Stuttgart

formationen übertragen werden, sondern auch eine Alarmanlage und eine Wegfahrsperre integriert werden, die per BLE aktiviert und deaktiviert werden kann. Unter der Information sollen sich der Ladezustand der Batterie, die Durchschnittsgeschwindigkeit und die gefahrene Strecke befinden. Eine einfache Erweiterung wäre es, die bereits vorhandenen GPS-Koordinaten, auf dem Endgerät zu speichern. Sobald das Fahrzeug geparkt wird, kann nun mit den GPS-Daten des Endgeräts die exakte Parkposition des Fahrzeuges ermittelt und wieder aufgefunden werden. Die Alarmanlage misst bei geparktem Fahrzeug die Bewegungen, die vom Fahrzeug ausgehen und aktiviert bei Überschreitung der Grenze, die bereits vorhandene Hupe, und lässt diese für eine bestimmte Zeit aktiv. Der zweite Sicherheitsas-

pekt ist die Wegfahrsperre, die mit Hilfe der Ladekontrolle dem Motorsteuergerät mitteilt, dass ein momentanes Losfahren des Fahrzeuges nicht möglich sein soll, darauf hin sperrt das Steuergerät den Fahrbetrieb. Somit bietet dies einen weiteren Diebstahlschutz zum bereits vorhandenen Schlüssel. Für den gesamten Aufbau wird ein programmierbares Steuergerät der Firma GIGATRONIK verwendet, welches eine BLE- und CAN-Schnittstelle besitzt, sowie Analogeingänge für den Batteriezustand und digitale Ausgänge zur Steuerung der Wegfahrsperre & Alarmanlage. Die GPS- und Sensor-Daten werden von einem frei programmierbaren GPS-Modul per CAN-Interface an das Steuergerät weitergegeben.

-
- [1] <http://www.navigantresearch.com/wordpress/wp-content/uploads/2011/06/NEV-11-Executive-Summary.pdf>
[2] <http://www.springerprofessional.de/die-mercedes-benz-vision-eines-golf-cart/4577638.html>
[3] <http://www.cityofpalmdesert.org/index.aspx?page=225>

Bildquellen:

- Abbildung 1: Mercedes Benz
- Abbildung 2: Betriebsanleitung Fahrzeuge mit Elektromotor - EZ-GO RXV

Konzeption und Implementierung eines grafischen Editors zur Skriptverwaltung einer bestehenden Softwarelösung

Christian Holder*, Manfred Dausmann, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Heutzutage werden die meisten Bauteile und Baugruppen zunächst digital konstruiert. In den letzten Jahren schreitet die Digitalisierung auch bei den Fertigungsprozessen immer weiter voran. Oft wird hier der Begriff der „digitalen Fabrik“ verwendet, welcher als Oberbegriff ein umfangreiches Netzwerk von digitalen Modellen und Methoden definiert [1]. Zweck solch einer Fabrik ist die ganzheitliche Planung, Umsetzung, Steuerung und Verbesserung aller dazugehörigen Prozesse. FAST-SUITE E2 der CENIT AG ist eine Softwarelösung für die digitale Fabrik, welche von der Roboter Offline Programmierung und Maschinen-Simulation bis hin zur Simulation kompletter Linien- und Produktionsanlagen Verwendung findet. Ein Vorteil der Nutzung einer solchen Lösung ist, dass durch die Simulation der Pro-

zesse, Fehler frühzeitig identifiziert und behoben werden können, die ansonsten bei der realen Umsetzung enorme Schäden verursachen könnten. Ein gutes Beispiel wäre hierfür das frühzeitige Erkennen von Kollisionen der Roboter, welche nun vor Inbetriebnahme beseitigt werden können. Nicht nur mögliche Fehler können behoben werden, sondern auch Optimierungen hinsichtlich der Bearbeitungszeiten können erzielt werden. Erst nach der erfolgreichen Simulation wird das somit erstellte Programm auf die beteiligten Roboter der Produktionsstätte übertragen. Daher verringert sich die Standzeit der Produktionsstätte, da diese während der Programmierung weiter im Betrieb ist.

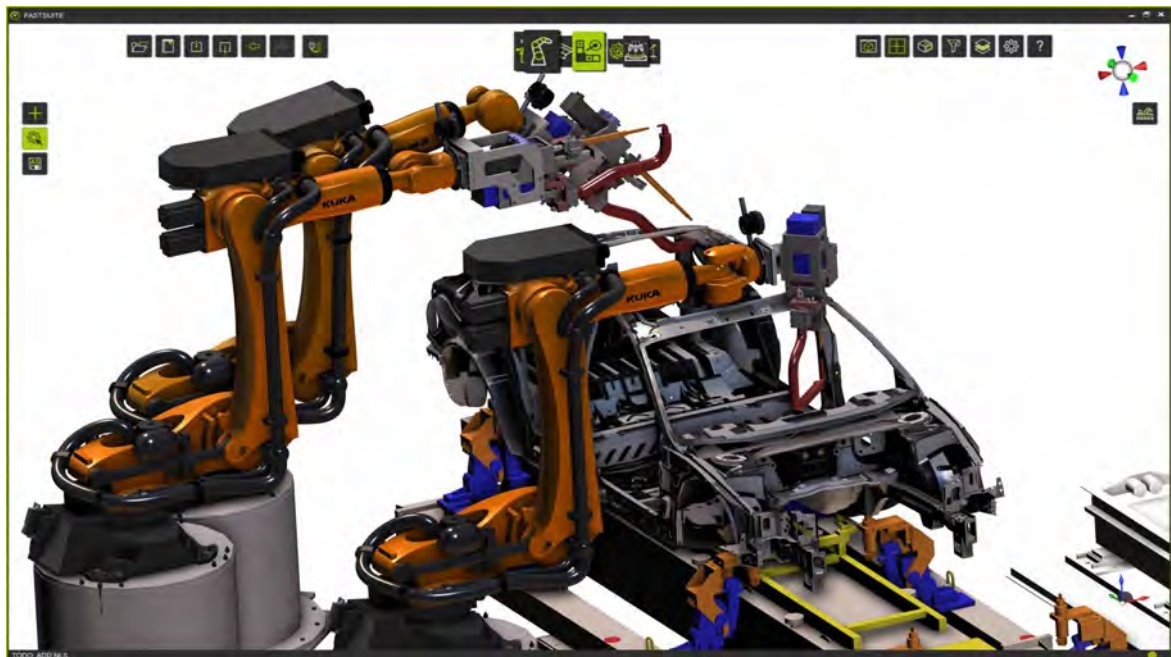


Abbildung 1: FASTSUITE E2

*Diese Arbeit wurde durchgeführt bei der Firma CENIT AG, Stuttgart

Im Rahmen der zu erstellenden Abschlussarbeit soll nun für die Softwarelösung FAST-SUITE Edition2 ein Konzept zur Verwaltung und Erstellung von Skripten erarbeitet werden. Diese Skripte stellen das Verhalten eines Maschinencontrollers dar. Im Genaueren bedeutet dies, dass zum Beispiel verschiedene Funktionsaufrufe stattfinden oder dass auch einfache Verzweigungen und Bedingungsabhängigkeiten, wie zum Beispiel das Abwarten eines Signales mithilfe des Skriptes beschrieben werden. Die Maschinencontroller verfügen je nach Hersteller, Bauart und diversen anderen Restriktionen einen begrenzten Funktionsatz, welcher bei der Erstellung und Verwaltung der Skripte berücksichtigt werden muss. Bisher werden diese Skripte mithilfe eines einfachen Texteditors erstellt. Um den Anwendern nun das Erstellen dieser Skripte zu vereinfachen soll ein grafischer Editor konzeptioniert und implementiert werden. Dieser Editor soll es ermöglichen, mithilfe eines Baukastens ein Ablaufdiagramm zu erstellen, damit das gewünschte Verhalten ohne viel direkter Texteingabe erreicht werden kann. Dieses Ablaufdiagramm wird in Form eines Flussdiagramms, auch bekannt als Programmablaufplan, angelegt. Im Anschluss soll aus diesem zusammengebauten Programmplan ein Skript in der gewünschten Skriptsprache erzeugt werden. Vorteil eines solchen Editors ist es, dass eine Skripterstellung seitens des Anwenders mög-

lich ist, ohne dabei die Skriptsprache selbst im Detail beherrschen zu müssen.

Zunächst soll im Rahmen der Bachelorarbeit nur Python als Skriptsprache unterstützt werden, jedoch soll bei der Konzeptionierung der Skripterstellung bereits berücksichtigt werden, dass eine Unterstützung von verschiedenen Skriptsprachen das angestrebte Ziel ist.

Nicht nur die Generierung von Skripten soll der grafische Editor bereitstellen, sondern auch die Verwaltung der erzeugten Controllerskripte. Im Detail bedeutet dies, dass nicht nur das aus dem Ablaufdiagramms entstandene Skript im FASTSUITE Edition2 hinterlegt sein soll, sondern auch das Diagramm selbst, falls eine nachträgliche Bearbeitung der Logik vonnöten ist. Das Auslesen und die Umwandlung zu einem Ablaufdiagramm aus extern geschriebenen Skripten werden in diesem Konzept nicht umgesetzt, lediglich die Möglichkeit die Lösung um dieses Feature zu erweitern, soll gewährleistet werden.

Der grafische Editor selbst ist mit WPF (Windows Presentation Foundation) zu realisieren, da die zu erweiternde Softwarelösung auf dieser Technologie basiert und daher eine einfachere Einbettung des Editors in die vorhandene Lösung gewährleistet sein soll. Erschwerend zur Entwicklung des Konzeptes kommt hinzu, dass das Konzept MVVM konform zu gestalten ist, damit es in die vorhandene Lösung eingebettet werden kann.

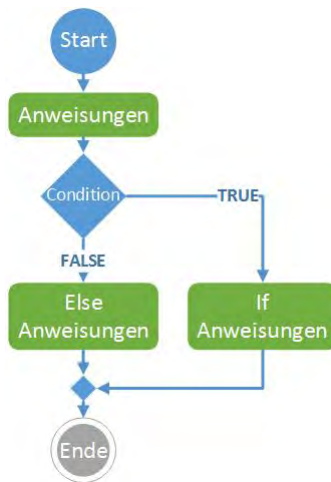


Abbildung 2: Beispiel eines Programmplans

[1] VDI-Richtlinie VDI 4499 Blatt 1:2008-02

Bildquellen:

- Abbildung 1: CENIT AG
- Abbildung 2: Selbst erstelltes Bild

Konzepterstellung zur sicheren Datenkommunikationsschnittstelle im Bereich Connected Cars

Tobias Horn*, Reinhard Schmidt, Dominik Schoop

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Das Internet of Things (IOT) ist das neue Schlagwort dieser Tage. Neben Industrieanlagen, Smartphones und intelligenten Drohnen zählen auch Fahrzeuge zu den wichtigsten Vertretern. Diese erhalten durch sogenannte Cloud Controlling Units (CCU) Zugang zum Internet. Dadurch können fahrzeuginterne Daten nach außen kommuniziert werden. Diese Schnittstelle zum Internet birgt jedoch auch viele Gefahren. So haben bereits einige findige Hacker erfolgreich unerwünschten Zugang zu diesen Daten bekommen und es sogar geschafft die Kontrolle über das Fahrzeug zu übernehmen [1].

Das Fahrzeug war seit jeher streng vom Rest der Welt abgekoppelt. Daher haben IT-Sicherheitsstandards, die in der Computerbranche seit Jahren als Stand der Technik gel-

ten, im Automobil nie Einzug erhalten. Nachdem nun durch die CCU das Fahrzeug ans Internet angebunden wird, klaffen große Sicherheitslücken. Deshalb muss das Sicherheitskonzept komplett überarbeitet und an die heutigen Anforderungen angepasst werden. Die Motivation dieser Arbeit ist darin begründet, ein Sicherheitskonzept für ein von der Robert Bosch GmbH öffentlich gefördertes Projekt zu erarbeiten. Dies beinhaltet die Kommunikation zwischen der CCU und den Steuergeräten im Fahrzeug, sowie die Kommunikation zwischen der CCU und dem Cloud Server. Dazu gehört eine Recherche über die Verfahren und Techniken mit denen Hacker die Fahrzeuge heutzutage angreifen. Außerdem soll eine übersichtliche und gut verständliche Darstellung für das Security-Diagramm gefunden werden.

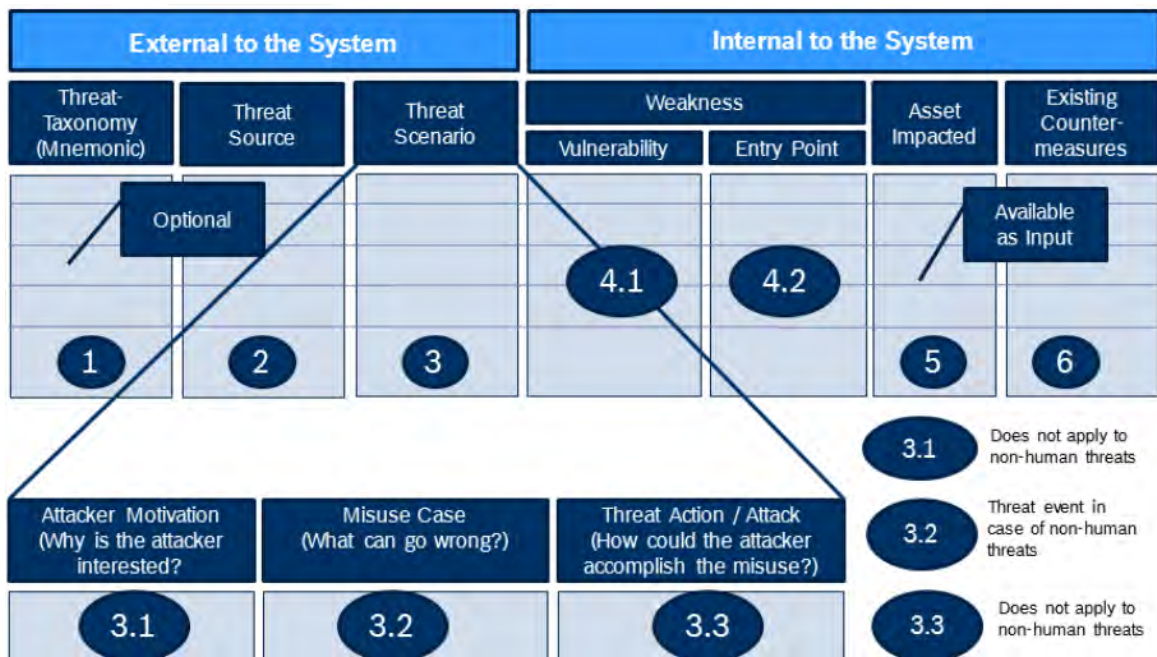


Abbildung 1: Ergebnis der Risiko und Bedrohungsanalyse

Grundlegend für die Konzeption ist die Systemarchitektur. Diese wurde durch das Tool Enterprise Architect 10 (EA10) dargestellt. Hieraus wurden, ebenfalls in EA10, ein Kom-

ponentendiagramm, ein Datenflussdiagramm sowie ein Kontrolldiagramm abgeleitet. Basierend auf den Diagrammen wird ein Security-Diagramm erarbeitet, welches auf einfacher

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Abstatt

und übersichtlicher weiße alle IT Sicherheitsrelevanten Informationen darstellt. Daraus sind zum Beispiel die bereits bestehenden Sicherheitsfunktionen und die möglichen Angriffspunkte klar ersichtlich. Auf Grundlage des Security-Diagramms ist es nun möglich, eine Gefahren- und Bedrohungsanalyse für das Projekt durchzuführen. Anhand dieser Analyse wird festgelegt, welche Risiken für das System tragbar sind und welche nicht. Hierbei werden zuerst der Umfang und die Tiefe des zu betrachtenden Systems festgelegt. Nachdem diese Punkte festgelegt sind, werden die möglichen Gefahrenquellen herausgearbeitet. Diese können Non-Human, Human-Malicious und Human-non-Malicious sein. Non-Human Gefahrenquellen beschreiben Gefahren, die nicht vom Menschen ausgehen sondern von dem System. Das inkludiert technische Mängel, Prozessfehler und Systemfehler. Human-Malicious Gefahrenquellen beschreiben Gefahren, die von Menschen ausgehen die gezielt und mit böswilliger Absicht dem System schaden wollen. Human-non-Malicious beschreibt Gefahren, die von Menschen ausgehen, die ungeplant und versehentlich Aktionen ausführen welche dem System schaden. Anschließend sind die Annahmen für das System und deren Komponenten zu ermitteln. Es muss außerdem hinterfragt werden, ob Rechte für bestimmte Benutzergruppen gerechtfertigt sind. Sogenannte „Third-Party Komponenten“, also Komponenten auf deren Sicherheit man keinen direkten Einfluss nehmen kann, müssen in ihrer Vertrauenswürdigkeit festgelegt werden. Nachdem das Gesamtsystem untersucht

worden ist, werden die einzelnen Komponenten genauer analysiert. Hierfür wird eine Taxonomie für die Gefahren erarbeitet. Bevor nun Gegenmaßnahmen identifiziert werden können werden bekannte Schwachstellen und bekannte Angriffe auf solche Systeme recherchiert und untersucht. Mit all diesen Informationen ist es nun möglich ein sogenanntes Threat Model zu erstellen, welches die Angreifer und deren Intentionen beschreibt. Es folgt die Risikobeurteilung. Hierfür werden die Konsequenzen eines Angriffs sowie die Wahrscheinlichkeit, dass dieser Angriff durchgeführt wird, festgelegt. Aus diesen beiden Informationen wird nun eine Risikomatrix erstellt.

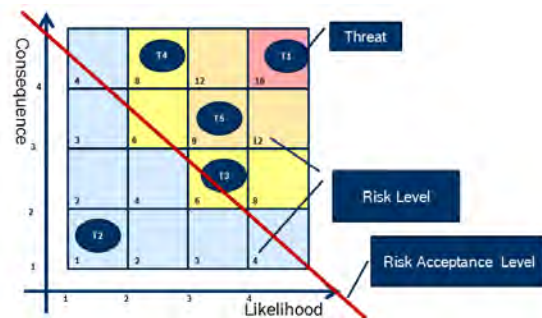


Abbildung 2: Risiko-Akzeptanz Diagramm

Dieses Ergebnis dient dann als Grundlage zur Auswahl nötiger und konkreter Sicherheitsmechanismen, welche die in dieser Arbeit erkannten Risiken abdecken.

[1] <http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/>

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Eigene Darstellung

Integration des Kommunikationsprotokolls DDS (Data Distribution Service) in einen flexiblen Protokollstapel für die Datenübertragung in Fahrzeugnetzwerken

Steffen Huzel*, Manfred Dausmann, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Verfügbarkeit von Komfort- und Fahrerassistenzfunktionen wächst in der Automobilbranche von Jahr zu Jahr. Sie sollen das Fahren komfortabler und vor allem sicherer machen. Diese Entwicklung hat jedoch eine Zunahme der Kommunikationsbeziehungen zwischen den im Fahrzeug verbauten Steuergeräten und der ausgetauschten Datenmenge zur Folge. Kamerabasierte Fahrerassistenzsysteme waren einer der Haupttreiber für die Einführung von Ethernet im Automobilbereich. Im Vergleich zu den vorhandenen Bussystemen wie CAN, LIN oder FlexRay bietet Ethernet den Vorteil von skalierbarer Bandbreite und einer hohen Flexibilität. Durch die Einführung der BroadR-Reach-Technologie, ein Ethernet-Standard im Automobilbereich, der eine Kommunikation über eine 2-Draht-Leitung ermöglicht, waren die anfänglichen Hürden in der Verkabelungstechnologie genommen. Das Internet-Protokoll und Ethernet als neue Netzwerktechnologie eröffnet den Ingenieuren der Automobilindustrie neue Türen für Innovationen.

Momentan findet ein Wandel von der bisher statisch konfigurierten Kommunikation hin zu einer dynamischen, service-orientierten Kommunikation statt. Dies ermöglicht eine flexible Vernetzung innerhalb des Fahrzeugs und über die Luftschnittstelle hinweg zu entfernten Kommunikationspartnern.



Abbildung 1: Das Middleware-Prinzip

Durch die neuen Möglichkeiten des Ethernets können nun auch Zwischenanwendungen, oft als „Middleware“ bezeichnet, eingesetzt werden. Als Middleware wird eine an-

wendungsneutrale Softwareschicht bezeichnet, die zwischen Anwendungen vermittelt und so die Komplexität der darunter liegenden Softwareschichten verbirgt. Eine Kommunikationsorientierte Middleware abstrahiert und verbirgt Netzwerk-Infrastruktur. Zur eigentlichen Kommunikation liegt oftmals ein speziell angepasstes Kommunikationsprotokoll vor, welches auf bestehende Protokollschichten aufsetzt. Mit SOME/IP wurde schon ein erstes Protokoll für die service-orientierte Ethernet-Kommunikation in AUTOSAR standardisiert [1].

Das „Data Distribution Service“ (DDS) ist ein Middleware-Protokoll, welches auf dem bestehenden Internet Protokoll (IP) aufbaut [2]. Es wurde durch die „Object Management Group“ (OMG) spezifiziert und verwendet ein Publish-Subscribe-Konzept. Dieses Konzept sorgt für eine Entkopplung von Sendern (Publisher) und Empfängern (Subscriber). So genannte Topics dienen als Informationsträger und werden in einem globalen Datenraum verwaltet. Sie bilden die zentrale Stelle für Sender und Empfänger. So kann eine beliebige Anzahl von Sendern oder Empfängern Daten in einem Topic schreiben (publish) oder diese abonnieren (subscribe). Abbildung 2 veranschaulicht diese Zusammenhänge.

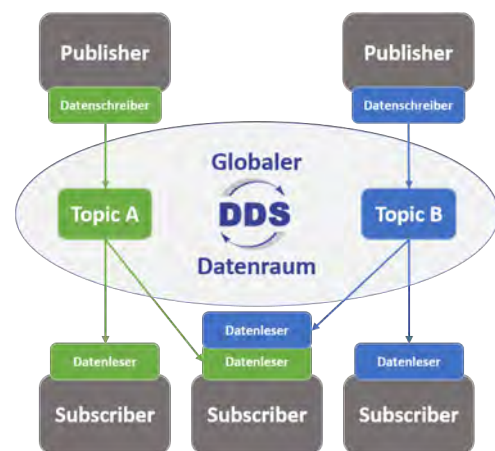


Abbildung 2: Das DDS-Publish-Subscribe-Prinzip

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Renningen

DDS bietet gegenüber SOME/IP eine breite Auswahl von „Quality of Service“- (QoS) und Security-Einstellungen auf Serviceebene. Dadurch können Datenbereiche gegen einen Ausfall oder gegen einen unerwünschten Zugriff geschützt werden, welches wichtige Kriterien sind für zukünftige Fahrerassistenzsysteme.

Ziel dieser Bachelorarbeit war die Anbindung der Middleware DDS an einen automotiv spezifischen Protokollstapel, der in der Lage ist über mehrere Kommunikationsprotokolle zu kommunizieren. Eine einheitliche Schnittstelle zur Anwendungsprogrammierung dient zur Anbindung verschiedener Anwendung an den Protokollstapel. Eine weitere Schicht übernimmt die Vermittlung zwischen den verschiedenen Kommunikationsprotokollen. Abbildung 3 zeigt diese Zusammenhänge innerhalb des Protokollstapels. Zu den Kernaufgaben der Arbeit gehörten das Design und die Implementierung eines Software-Moduls, das eine bestehende DDS-Implementierung in den Protokollstapel integriert. Hierfür wurde eine

kommerzielle DDS-Implementierung, die bereits in verschiedenen Branchen zum Einsatz kommt, verwendet. Über eine Funktionsabbildung der DDS-Funktionalitäten auf die Vermittlungsschicht, wurde die Protokollanbindung entworfen. Unter anderem musste die Servicesuche des DDS-Protokolls auf die der Vermittlungsschicht abgebildet werden. Des Weiteren wurden erste QoS-Einstellungen im Software-Modul umgesetzt. Sie sorgen dafür, dass Daten die während eines Verbindungsabbruchs nicht zugestellt werden können, nach einem Wiederaufbau der Verbindung zugestellt werden. Nach der erfolgreichen Implementierung des Software-Moduls, in der Programmiersprache C++, wurde ein Demonstrator aufgebaut. Die Umsetzung des Demonstrators erfolgte auf der Hardware zweier Raspberry Pi und einem Switch. Ein Raspberry Pi besitzt dabei jeweils eine Anwendung mit je einem Publisher und einem Subscriber. Die übertragenen Daten werden auf einem kleinen, aufsteckbaren Display angezeigt.

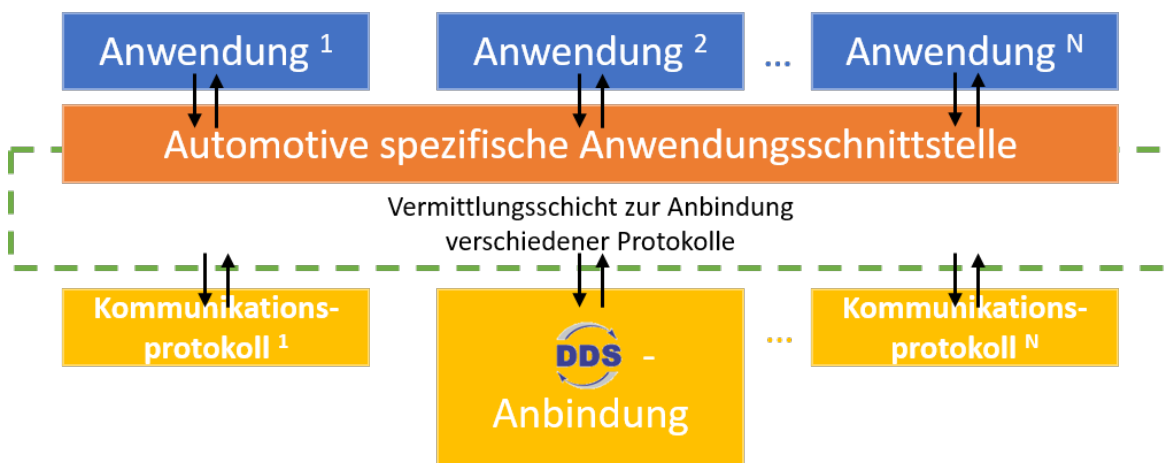


Abbildung 3: Kommunikationsstapel zur Anbindung verschiedener Kommunikationsprotokolle

[1] SOME/IP: <http://some-ip.com/>

[2] DDS: <http://portals.omg.org/dds/>

Bildquellen:

- Abbildung 1–3: Eigene Darstellung

Modellbildung zur echtzeitfähigen und beobachterbasierten Temperaturüberwachung für den Zwischenkreiskondensator eines Pulswechselrichters für Elektro- und Hybridfahrzeuge

Neşe Inan*, Walter Lindermeir

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Elektromobilität gewinnt immer mehr an Bedeutung, vor allem durch den Klimawandel und durch die Auflagen der Europäischen Union zur Verminderung von Kohlenstoffdioxidemissionen [1]. Elektro- und Hybridfahrzeuge stoßen keinen bzw. weniger CO_2 aus als konventionell angetriebene Fahrzeuge. Zusätzlich ist zu bedenken, dass die fossilen Ressourcen der Erde schlussendlich limitiert sind, so dass früher oder später nach alternativen Antrieben gesucht werden muss. Darüber hinaus haben Elektrofahrzeuge den Vorteil, dass sie deutlich leiser sind und somit die Lärmbelastigung senken. Jedoch befindet sich die Verbreitung von Elektro- und Hybridfahrzeugen noch in der Anfangsphase. So wurde am 1. Januar 2015 ein bundesweiter Bestand von 18.948 Elektrofahrzeugen und von 107.745 Hybridfahrzeugen gemeldet [2]. Dies entspricht einem Anteil von lediglich 0,043 % bzw. 0,243 % an der Gesamtanzahl der Pkws in Deutschland.

Der elektrische Antriebsstrang setzt sich zusammen aus der Hochvoltbatterie, die den Gleichstrom zur Verfügung stellt, dem Pulswechselrichter und der elektrischen Maschine, die mit Dreiphasenwechselstrom betrieben wird. Der Pulswechselrichter stellt das Stellglied für die Drehzahl- und Drehmomentregelung der elektrischen Maschine dar. Er wandelt die Gleichspannung in eine Dreiphasenwechselspannung um, indem er mithilfe eines Pulsverfahrens den Ausgangsströmen einen sinusförmigen Verlauf verleiht, was jedoch zu einem pulsierenden Gleichstrom im Zwischenkreis führt. Der Zwischenkreiskondensator dient als Puffer, so dass zum einen die Hochvoltbatterie geschont wird und zum anderen der Einfluss der Leitungsimpedanzen vermindert wird, so dass der Strom so schnell wie nötig zur Verfügung steht.

Den hohen Leistungen im Antriebsstrang steht die kompakte Bauweise der leistungselektronischen Bauteile gegenüber, was zu hoher Verlustleistungsdichte und höherer Bauteilerwär-

mung führt. Falls die kritische Temperatur des Bauteils evtl. sogar überschritten wird, wird riskiert, dass das Bauteil schneller altert oder sogar beschädigt wird. Daher ist es notwendig die Temperatur des entsprechenden Bauteils in Echtzeit überwachen zu können. Je präziser die Temperaturschätzung ist, desto später ist eine Abregelung möglich, was zu einer höheren Leistungsverfügbarkeit führt.

Im Rahmen dieser Arbeit soll ein echtzeitfähiges thermisches Modell für die Temperaturüberwachung des Zwischenkreiskondensators in einem Pulswechselrichter entwickelt werden. Für die Entwicklung des Modells wurde zunächst untersucht, welche Verlustleistungseinträge am Kondensator eine Rolle spielen. Im Gegensatz zum idealen Kondensator, bei dem keine Verlustleistung entsteht, da er lediglich einen Blindwiderstand darstellt, entstehen beim realen Kondensator Verluste. Ein Ersatzschaltbild des realen Kondensators ist in Abbildung 1 dargestellt und setzt sich zusammen aus dem frequenzabhängigen ESR (equivalent series resistance), dem ESL (equivalent series inductance) und der eigentlichen Kapazität C des Kondensators.

Einen großen Anteil beschreibt die Verlustleistung, die durch den pulsierenden Gleichstrom im Zwischenkreis am ESR entsteht.

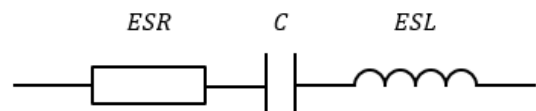


Abbildung 1: Ein Ersatzschaltbild eines Kondensators

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Tamm

Diese Verlustleistung P_L hängt vom Frequenzspektrum des Zwischenkreisstroms ab. Mit Hilfe der (messtechnisch ermittelten) ESR-Kennlinie und den Effektivströmen der einzelnen Frequenzanteile $I_{RMS}(f)$, kann diese gemäß

$$P_L = \sum_f ESR(f) \cdot I_{RMS}^2(f) \quad (1)$$

berechnet werden.

Für die Entwicklung des thermischen Modells soll zunächst anhand von diversen Messungen untersucht werden, welche Stellen des Zwischenkreiskondensators überhaupt kritisch hinsichtlich der Temperatur sind. Die Modellierung basiert auf der Erstellung eines thermischen Netzwerks. Zwischen der Thermodynamik und der Elektrotechnik existieren Analogien, die vor allem die Veranschaulichung der thermischen Zusammenhänge vereinfachen. Verlustleistungen können als Stromquellen, Temperaturen als Spannungen, Wärmekapazitäten als Kondensatoren und Wärmewiderstände als elektrische Widerstände ausgedrückt werden. So kann das thermische Verhalten wie ein elektrisches Schaltbild beschrieben werden:

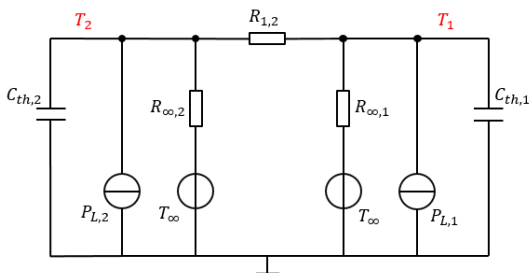


Abbildung 2: Beispielhaftes thermisches Netzwerk mit den Knoten T_1 und T_2

Wenn ein Knotenmodell steht, gilt es die Wärmeleitwerte und Wärmekapazitäten zu bestimm-

men. Dafür wird der Zusammenhang

$$\dot{T} = \frac{\dot{Q}}{C_{th}} \quad (2)$$

benutzt, der besagt, dass der Temperaturgradient \dot{T} sich aus dem Wärmestrom \dot{Q} und der Wärmekapazität C_{th} berechnet, wobei der Wärmestrom entweder durch Verlustleistung oder durch den Wärmefluss von oder zu anderen Knoten entsteht. Für jeden Knoten wird eine Differentialgleichung aufgestellt und die thermischen Zeitkonstanten identifiziert. Die Parameteridentifikation wird in dieser Arbeit mithilfe von Messdaten und einer Ausgleichsrechnung mit der linearen Least-Squares-Methode durchgeführt. Anschließend wird zur Überprüfung der Identifikation das Modell mit den gefitteten Parametern simuliert und mit den Original-Messungen verglichen.

Bei ausreichender Modellgenauigkeit des thermischen Netzwerks, soll im nächsten Schritt ein Störgrößenbeobachter entworfen werden. Es soll untersucht werden, ob die Umgebungstemperatur mithilfe des Störgrößenbeobachters geschätzt werden kann.

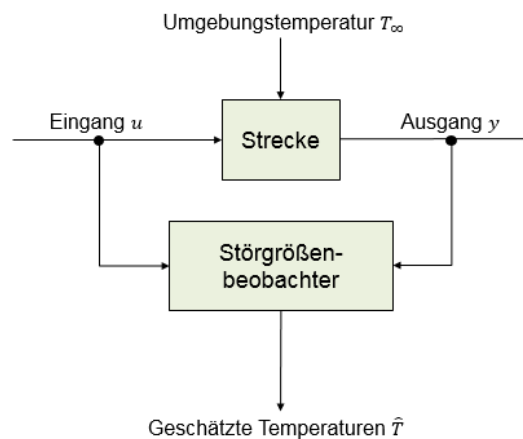


Abbildung 3: Strecke mit Störgrößenbeobachter

-
- [1] Umweltbundesamt: „Internationale und EU-Klimapolitik“, unter <http://www.umweltbundesamt.de/themen/klimaenergie/internationale-eu-klimapolitik> (abgerufen am 23.11.2015)
- [2] Kraftfahrtbundesamt: „Bestand an Pkw am 1. Januar 2015 nach ausgewählten Kraftstoffarten“, unter http://www.kba.de/DE/Statistik/Fahrzeuge/Bestand/Umwelt/2014_b_umwelt_dusl_absolut.html (abgerufen am 22.11.2015)

Das Architekturmuster Model-View-ViewModel

Lukas Jaeckle*, Joachim Goll, Manfred Dausmann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Model-View-ViewModel (Kurzform: MVVM) ist ein Architekturmuster für interaktive Systeme. Als solches dient es Entwicklern als Vorlage, um grafische Oberflächen standardisiert und strukturiert zu implementieren. MVVM kann als eine Weiterentwicklung bewährter Architekturmuster wie das Model-View-Controller-Muster (Kurzform: MVC) betrachtet werden.

MVVM entstand 2005 durch die Entwicklung des Windows Presentation Foundation Frameworks (Kurzform: WPF) und dem .NET Framework 3.0. Darüber hinaus ist MVVM fester Bestandteil von Silverlight, den neuen Universal Apps, aber auch von gegenwärtig beliebten Webframeworks wie KnockoutJS oder AngularJS. Ferner werden auch existierende Frameworks wie JavaFX durch MVVM erweitert (mvvmFX).

Wie ist MVVM aufgebaut? Wo liegen die Vorteile, aber auch, welcher Voraussetzungen bedarf es für MVVM? Was macht MVVM anders als das weitverbreitete MVC? Diese Fragen sollen nachfolgend beantwortet werden.

Aufbau von MVVM

Interaktive Systeme müssen mehrere Aufgaben erfüllen. Die Offensichtlichste ist die **Darstellung** einer Benutzeroberfläche. Benutzereingaben, die über diese Oberfläche getätigt werden, müssen des Weiteren entsprechend entgegengenommen und verarbeitet werden (**Eingabeverarbeitung**). Neben diesen interaktiven Aufgaben muss eine Anbindung an das weitere System hergestellt werden. Dieses besteht zum einen aus einer **Datenhaltung** und zum anderen aus den systemweit gültigen **Verarbeitungsprozessen**. Insbesondere bei verteilten Systemen werden die Verarbeitungsprozesse oftmals getrennt betrachtet, sodass die interaktiven Aufgaben auf Darstellung, Eingabeverarbeitung und einer reduzierten Datenhaltung für die Darstellung beschränkt werden können. Diese Aufgaben werden in MVVM auf klar abgegrenzte Komponenten verteilt (siehe Abbildung 1).

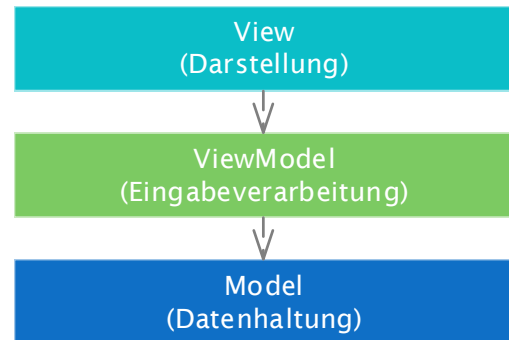


Abbildung 1: Komponenten und Abhängigkeiten

In der **View** wird die Benutzeroberfläche eines Systems festgelegt. In WPF wird dazu die deklarative Sprache XAML verwendet. Im XML-Format kann dadurch u. a. die Anordnung von Kontrollelementen spezifiziert werden. Dazu sind keine Programmierkenntnisse vonnöten, wodurch **Oberflächendesigner leichter** in die Entwicklung **eingebunden** werden können.

Das **ViewModel**, gerne auch als Model der View bezeichnet, beinhaltet Datenfelder und die Eingabeverarbeitung. Die Datenfelder des ViewModel können von der View an deren Kontrollelemente gebunden werden, sodass z. B. eine TextBox (**View**) die **Daten aus** einem Datenfeld des **ViewModel** anzeigt bzw. Benutzereingaben direkt in ein Datenfeld des ViewModel übernehmen kann. Ähnlich verhält es sich bei der Eingabeverarbeitung. So kann z. B. ein Button der View an eine Eingabeverarbeitung des ViewModel gebunden werden, sodass ein Klick automatisch die Eingabeverarbeitung ausführt. Soll dies temporär nicht möglich sein, kann das **ViewModel** den Zustand der Eingabeverarbeitung verändern und **die View** darüber **informieren**, sodass die View ihre Kontrollelemente entsprechend des Zustandes de- bzw. aktiviert.

Wie das **Model** in MVC ist das Model in der ursprünglichen Definition des MVVM-Musters verantwortlich für die Geschäftslogik und die Datenhaltung [1]. In neueren Interpretationen besteht das Model nur aus **Entity-Objekten**. Die Geschäftslogik wird dabei in einer separa-

*Diese Arbeit wurde durchgeführt bei der Firma IT-Designers, Esslingen-Zell

ten Komponente implementiert. Letzteres bildet die Grundlage für eine gute Umsetzung von Separation-Of-Concerns.

MVVM in WPF

Wie geschildert, liegt der große Vorteil von MVVM in der **bidirektionalen Synchronisati-**

on zwischen View und ViewModel. Diese Synchronisation muss jedoch durch das Framework unterstützt werden. WPF nutzt dafür das **Data Binding** und die **Änderungsbenachrichtigung** des .NET Frameworks. Beispielhaft ist dies nachfolgend dargestellt.

```
<Window x:Class="HelloWorld.View.HelloWorldV" ...>
  <Grid> <TextBox Text="{Binding TextBox}" /> </Grid> </Window>

public class HelloWorldVm : INotifyPropertyChanged {
  public event PropertyChangedEventHandler PropertyChanged;
  protected virtual void OnPropertyChanged([CallerMemberName] string propertyName=""){
    PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));
  }

  private string _textBox = "Hello World";
  public string TextBox { get { return _textBox; }
    set { _textBox = value; OnPropertyChanged(); } }
}
```

Codebeispiel 1: Data Binding WPF

In der View wird im XML-Format angegeben, welches Kontrollelement welche Properties bzw. Commands des ViewModel nutzen soll (Data Binding). Obwohl sich die View dadurch abhängig von dem ViewModel macht, kann das Layout bzw. die **View nahezu unabhängig vom ViewModel entwickelt** werden.

Das ViewModel muss die benötigten Properties und Commands bereitstellen und eine Änderungsbenachrichtigung implementieren. WPF bietet diesbezüglich das Interface `INotifyPropertyChanged` an. Dieses Interface erfordert die Implementierung eines Events, welches bei Wertänderungen ausgelöst werden muss. Dies ist in dem Codebeispiel 1 angedeutet.

Der Vorteil dieser Lösung ist, dass das ViewModel unabhängig von der View bleibt. Dadurch kann das **ViewModel sehr gut getestet** werden. Zudem bietet das Eventhandling den Vorteil, dass jeweils nur **einzelne Elemente synchronisiert** werden. Der Overhead bleibt also gering.

MVVM im Vergleich zu MVC

Worin liegt folglich der Unterschied zu MVC? In MVC holt sich die View über das Observable-Muster direkt von dem Model die

Daten ab. Dadurch besitzt MVC eine zusätzliche **Abhängigkeit zwischen der View und dem Model**. MVVM verschiebt die Datenbereitstellung für die View in das ViewModel, wodurch diese Abhängigkeit **entfällt**.

Der zweite Unterschied liegt darin, dass das **ViewModel keine Abhängigkeit zur View** besitzt. In MVC besitzt der Controller eine Referenz auf die View, damit der Controller den Zustand von Kontrollelementen unmittelbar setzen kann. Dies erschwert das Erstellen von Tests allerdings deutlich. MVVM hingegen, ermöglicht durch die bidirektionale Synchronisation zwischen der View und dem ViewModel, dass der Zustand der View im ViewModel verwaltet werden kann, ohne eine Referenz auf die View zu benötigen. Durch diese **lose Kopplung** ist die Testbarkeit verbessert.

Bewertung

MVVM in Kombination mit WPF bietet **Entwicklern und Oberflächendesignern** eine gute Ausgangsbasis, um interaktive Systeme **klar und strukturiert** zu entwickeln. Durch **wenige Abhängigkeiten** wird zudem eine **gute Wartbarkeit** der Komponenten ermöglicht.

-
- [1] J. Gossman (08. Okt 2005). Introduction to Model/View/ViewModel pattern for building WPF apps: <http://blogs.msdn.com/b/johngossman/archive/2005/10/08/478683.aspx>, Zugriff am 7. Aug 2015.
 [2] J. Goll (2014). Architektur- und Entwurfsmuster der Softwaretechnik, 2. Auflage, Springer Vieweg, pp. 377-390.

Stabile und evolvierbare Software durch Einhaltung der SOLID-Prinzipien

Markus Just*, Manfred Dausmann, Joachim Goll

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Softwaresysteme, die heutzutage entwickelt werden, sind sehr komplex. Bei ihrer Entwicklung muss alles getan werden, um Fehler zu vermeiden. Dies betrifft nicht nur die Programmierung, bei der man „defensiv“ programmieren sollte, um das Auftreten von Fehlern möglichst zu verhindern. Auch der Entwurf eines Systems wird von Prinzipien geprägt, die eingehalten werden sollten, um ein stabiles Design zu erreichen.

Entwurfsprinzipien verkörpern Erfahrung und gelten allgemein. Sie werden außer in der selbst zu erstellenden Software auch in Entwurfsmustern eingesetzt. Entwurfsprinzipien gewährleisten, dass eine Software gewisse Qualitätsmerkmale besitzt. So ist z. B. „Programmieren gegen eine Schnittstelle“ ein Entwurfsprinzip, welches die Erweiterbarkeit und die Testbarkeit unterstützt. Aus diesem Grund wird dieses Prinzip auch in wiederholtem Maße bei Entwurfsmustern eingesetzt. Erweiterbarkeit durch Spezialisierung wird durch Einhalten des Liskovschen Substitutionsprinzips sichergestellt.

Aufgrund verschiedener Rahmenbedingungen wie Zeitdruck, Unerfahrenheit oder Nachlässigkeit werden Entwurfsprinzipien häufig verletzt. Um dies zu verhindern, ist es unerlässlich, den erstellten Quellcode manuell durch regelmäßige Reviews und durch die Nutzung diverser Tools zu prüfen.

Entwurfsziele für Entwurfsprinzipien

Durch den Einsatz von Entwurfsprinzipien sollen verschiedene Entwurfsmerkmale erreicht werden, die eine hohe Softwarequalität sicherstellen. Ziel ist es beispielsweise, **Verantwortlichkeiten der Softwaremodule zu trennen**, um Abhängigkeiten zu reduzieren. Außerdem soll die **Testbarkeit**, z. B. durch das Ermöglichen von Mocks, sowie **Korrektheit** gewährleistet sein. Auch **Evolvierbarkeit** spielt eine große Rolle. Evolvierbarkeit bedeutet nicht nur, dass das System **erweiterbar** ist, sondern dass die verschiedenen Komponenten auch im Rahmen anderer Architekturen **wiederverwendbar** sind.

Abhängigkeiten

Das Hauptproblem von Software ist in der Re-

gel das Vorhandensein zu vieler Abhängigkeiten innerhalb eines Systems. Daher gilt es, diese Abhängigkeiten, soweit es möglich ist, zu eliminieren, was das Entstehen verworrener Strukturen und einer starken Kopplung der Komponenten verhindert. Da es nicht möglich ist, Abhängigkeiten komplett zu entfernen, muss es das Ziel beim Entwurf eines Systems sein, diese so gering wie möglich zu halten. Abhängigkeiten können innerhalb eines selbst erstellten Softwaresystems auftreten oder auch in Bezug auf die Umgebung, in der das System läuft. Die Abhängigkeit zwischen verschiedenen Komponenten wird auch als Kopplung dieser Komponenten bezeichnet. Wenn diese in einem System mit hoher Kopplung ausgetauscht werden, ist die Software möglicherweise nicht mehr lauffähig.

Die SOLID-Prinzipien

Robert C. Martin fasst eine wichtige Gruppe von Prinzipien zur Erzeugung wartbarer und erweiterbarer Software unter dem Begriff SOLID-Prinzipien zusammen [1]. Dieser Begriff soll andeuten, dass die Verwendung dieser Prinzipien für das Schreiben hochwertiger Software unabdingbar ist.

SOLID steht für Single-Responsibility-Prinzip (SRP), Open-Closed-Prinzip (OCP), Liskovsches Substitutionsprinzip (LSP), Interface-Segregation-Prinzip (ISP) und Dependency-Inversion-Prinzip (DIP). Robert C. Martin erklärte diese Prinzipien zu den wichtigsten Entwurfsprinzipien. Die SOLID-Prinzipien selbst stammen aber nicht alle von Robert C. Martin, sondern im Falle des OCP von Bertrand Meyer und im Falle des LSP von Barbara Liskov. Im Folgenden werden die Bedeutungen und die zu erreichenden Entwurfsmerkmale der Prinzipien kurz vorgestellt.

Single Responsibility Prinzip

Das Single Responsibility Prinzip fordert, wie dem Namen zu entnehmen ist, dass Software-Module nur eine einzige Verantwortlichkeit realisieren, also nur eine einzige Aufgabe erfüllen sollen. Verantwortlichkeit steht in diesem Kontext für den Grund einer Änderung. Kann es also in einem Software-Modul meh-

*Diese Arbeit wurde durchgeführt bei der Firma IT-Designers GmbH, Esslingen

rere Gründe für eine Änderung geben, dann wird das SRP verletzt. Es soll eine **Trennung der Verantwortlichkeiten** und damit eine Reduktion der Abhängigkeiten erreicht werden. Das Single Responsibility Prinzip ist zwar ein einfach zu verstehendes Prinzip, wird in der Realität aber mit am häufigsten verletzt.

Open Closed Prinzip

Das Open Closed Prinzip fordert, dass Module offen für Erweiterungen und gleichzeitig geschlossen für Veränderungen sind. Offen für Erweiterungen bedeutet, dass ein Software-Modul durch das Hinzufügen von Code erweitert werden kann, was durch die Nutzung von Abstraktionen erreicht werden kann. Geschlossenheit eines Moduls gegenüber Veränderungen heißt, dass ein Modul wiederverwendet werden kann, ohne den Code und die vorhandenen Schnittstellen anpassen zu müssen. Das Einhalten dieses Prinzips trägt zur **Wiederverwendbarkeit** und zur **Erweiterbarkeit** bei.

Liskovsches Substitutionsprinzip

Das Liskovsche Substitutionsprinzip fordert von abgeleiteten Klassen im objektorientierten Entwurf, dass sie dasselbe Verhalten aufweisen wie ihre Basisklassen. Ein Client soll es also nicht merken, wenn an die Stelle des Objekts einer Basisklasse plötzlich ein Objekt einer abgeleiteten Klasse tritt. Dies kann auf verschiedenen Weisen realisiert werden. Bei einer reinen Erweiterung werden die Methoden einer Basisklasse nicht durch die abgeleiteten Klassen überschrieben. Gegenüber einem Client, der die Methoden der Basisklasse über ein Objekt der abgeleiteten Klasse nutzt, verhält sich ein Objekt der abgeleiteten Klasse genauso wie ein Objekt der Basisklasse. Sollte eine abgeleitete Klasse die Methoden ihrer Basisklasse überschreiben, müssen die Verträge der Basisklasse, also Vor- und Nachbedingungen der Methoden und Invarianten der Klasse, eingehalten werden, damit keine unerwünschten Nebeneffekte auftreten. Das LSP führt zu **Erweiterbarkeit** und **Korrektheit**.

Interface Segregation Prinzip

Das Interface Segregation Prinzip gibt vor, dass Interfaces ihren Clients nur Methodenschnittstellen anbieten sollen, die von diesen wirklich benötigt werden. "Fat" bzw. "wide"-Interfaces, das sind Interfaces, die eine hohe Anzahl an Methodenschnittstellen aufweisen,

sollen also vermieden werden. Sollte ein Interface Methodenschnittstellen aufweisen, die nicht von allen Clients benötigt werden, dann muss analysiert werden, wie man die Interfaces sinnvoll aufteilen kann, so dass jeder Client nur die Schnittstellen bekommt, die er auch benötigt. Somit wirken sich Änderungen an diesen Schnittstellen nur auf die Clients aus, die auch davon Gebrauch machen, also werden unnötige **Abhängigkeiten** vermieden.

Dependency Inversion Prinzip

Das Dependency Inversion Prinzip fordert, dass Klassen einer höheren Ebene nicht direkt von Klassen niedriger Ebenen abhängen sollen, sondern beide sollen von Abstraktionen abhängen. Oftmals werden Klassen in Ebenen eingeteilt. Dabei beschreibt in einem hierarchischen Modell nach Grady Booch die oberste Ebene die Politik der Geschäftsprozesse, die Ebene darunter beschreibt die Mechanismen und die unterste beschreibt Hilfsdienste. Anstatt des direkten Zugriffs und damit der direkten Abhängigkeit einer Klasse der höheren Ebene von einer Klasse der niedrigeren Ebene soll eine Abstraktionsschicht eingeführt werden, damit die Klasse der höheren Ebene nur noch von einer Abstraktion der unteren Ebene abhängt. Dieses Prinzip führt zu einer **Reduktion von Abhängigkeiten** und zu einer **Erhöhung der Testbarkeit**, indem es die Möglichkeit bereitstellt, Mocks zu nutzen.

Bewertung

SOLID legt die Basis für das Design eines robusten, wartbaren, korrekten und langfristig erweiterbaren Softwaresystems. Damit kann die Lebensdauer und Wiederverwendbarkeit von Software erhöht werden. Software, die nach diesen Prinzipien entwickelt wird, lebt nicht nur länger, sondern ist auch infolge der Reduktion von Abhängigkeiten leicht testbar. Es wird in mehrfachem Sinne durch SOLID Qualität erzeugt: Das Design verhindert das Altern der Software durch die Reduktion der Abhängigkeiten (Single Responsibility Principle, Interface Segregation Principle, Dependency Inversion Principle), erhöht die Testbarkeit durch Mocks (Dependency Inversion Principle), sorgt für Korrektheit (Liskovsches Substitutionsprinzip) und sorgt durch das Open-Closed-Prinzip für Stabilität und Erweiterbarkeit.

[1] Robert C. Martin: Agile Software Development, Pearson Ptr, 2011

Konzeption und prototypischer Aufbau eines mobilen Prüfsystems zur automatisierten Funktionsanalyse von Automotive-Aktuatorik-Komponenten

Daniel Kahrizi*, Karlheinz Höfer, Werner Zimmermann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Gewährleistung der Zuverlässigkeit und Sicherheit ist bei der Automobilentwicklung von entscheidender Bedeutung. Das Prüfen, Testen und Evaluieren sämtlicher Komponenten ist entsprechend Alltag bei der Entwicklung im Automobilbereich. Dies gilt natürlich auch für die verschiedenen Aktuatorikkomponenten wie Drosselklappen, Ventile, Heizer oder Pumpen. Diese Komponenten müssen in den verschiedenen Stadien der Entwicklung immer wieder sowohl kurzen aber aussagekräftigen Funktionstests, als auch ausführlichen und stichhaltigen Dauertests unterzogen werden. Das Ziel dieser Abschlussarbeit besteht in der Anforderungsanalyse, der Konzeptbildung sowie Prototypenerstellung eines Automotive-Aktuatoren-Prüfsystems, welches die Ingenieure bei den verschiedenen Aktua-

torprüfungen unterstützt.

Bei der Anforderungsanalyse steht zunächst die automatisierte Funktionsprüfung und das Aufnehmen von Kennlinien für die unterschiedlichen Aktuatoren in verbautem Zustand im Fahrzeug und in ausgebautem Zustand im Vordergrund. Der Mobilitätsanspruch der sich hieraus für das Prüfsystem ergibt, zieht Konsequenzen für die möglichen Messungen und die dabei verwendeten Messmethoden nach sich. Beispielsweise muss eine Funktionsprüfung eines Ventils auch ohne das Vorhandensein externer Hardware, wie Vakuumpumpen oder Durchflussmessern, möglich sein. Im Rahmen der Abschlussarbeit müssen hierfür alternative Messmethoden für die verschiedenen Aktuatoren (siehe Abbildung 1) gefunden werden.



Abbildung 1: Kategorisierung der Aktuatoren

Das Prüfsystem wird möglichst modular und flexibel aufgebaut, sodass Aktuatoren, welche nicht bei der Anforderungsanalyse berücksichtigt wurden, trotzdem ohne großen Aufwand vom Prüfsystem getestet werden können. Eine Kategorisierung der Aktuatoren und deren Prüfungen ermöglicht zudem ein einfaches Integrieren von Aktuatoren, welche ähnlich zu den bereits bekannten Aktuatoren sind. Zusätzlich zu den Messaufgaben muss das Prüfsystem auch die Sicherheit des angeschlossenen Aktuators, des Prüfers und

des Systems selbst gewährleisten. Dies wird durch Selbsttestfunktionen und eine ständige Überwachung der Randbedingungen bei der Prüfung umgesetzt. Werden zuvor festgelegte Grenzwerte überschritten oder erkennt der Prüfer eine Gefahr, so kann das System mit einer Notausfunktion in einen sicheren Zustand übergehen. Zusätzlich zur Funktionsprüfung wird bei der Anforderungsanalyse Rücksicht auf die Erweiterung des Systems um aufwendigere Dauertests genommen. Diese benötigen meist kontrollierte Umweltbedingungen und

*Diese Arbeit wurde durchgeführt bei der Firma IAV GmbH, Weissach

somit weitere externe Hardware, wie Vakuumpumpen, weiteren Sensoren oder gar Klimaschränke. Im Rahmen der Modularität des Systems soll eine Integration weiterer externer Hardware ohne großen Aufwand möglich sein.

Das Konzept, welches sich aus der zuvor durchgeführten Anforderungsanalyse ergibt, sieht mehrere Module vor. Von zentraler Bedeutung sind hierbei das DEWESoft Messgerät Sirius und die Hardware-Module des automatisierten Testsystems.

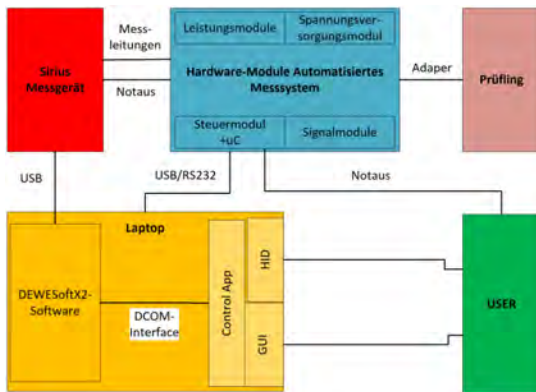


Abbildung 2: Übersicht der Module des Prüfsystems

Das Sirius-Messgerät verfügt über 8 analoge Spannungseingänge mit jeweils 1 MS/s, 8 Funktionsgeneratorausgänge und 8 digitalen Ein- und Ausgänge [1]. Die durch die Prüfungen geforderten Randbedingungen, wie beispielsweise eine Vierquadrantenansteuerung der Drosselklappe oder die hohen elektrischen Leistungsanforderungen an den Funktionsgenerator (max. 6A, 16V) können vom Messgerät alleine nicht erfüllt werden. Zudem sind für die Abdeckung aller Testfälle sehr unterschiedliche Messkonfigurationen notwendig. Aus diesen Gründen werden die Module des Blocks „Hardware-Module Automatisiertes Messsystem“ benötigt. Dies sind im Wesentlichen ein Spannungsversorgungsmodul, mehrere Leistungs- und Signalmodule und ein Steuermodul. **Spannungsversorgungsmodul:** Das Spannungsversorgungsmodul dient zur Bereitstellung der intern benötigten Spannungen, bei der Nutzung des Prüfsystems im Fahrzeug (Versorgungsspannung 12 V DC) und bei Netzspannung. **Leistungsmodul:** Die unterschiedlichen Aktuatoren müssen bei den Prüfungen unterschiedlich angesteuert werden

(gepulst, DC, AC mit DC-Offset, etc.). Beispielsweise kann durch den Wert der Spuleninduktivität eines Magnetventils Rückschluss auf die aktuelle Schaltstellung des Ventils gezogen werden. Für die Bestimmung dieser Induktivität muss ein DC-Offset behaftetes Sinus-Signal ausgegeben werden und gleichzeitig der Strom gemessen werden. Aus der Phasenverschiebung zwischen Spannung und Strom lässt sich anschließend die Induktivität berechnen. Da der Funktionsgenerator für diese Prüfung nicht ausreichend Leistung bereitstellen kann, muss das Signal verzerrungsarm verstärkt werden. Entsprechend verfügt das Modul über eine zuschaltbare Strommessfunktion und einen Verstärker. Zudem befindet sich auf dem Leistungsmodul eine zuschaltbare H-Brücke, mit welcher auch größere Lasten mit pulsweitenmodulierten Signalen angesteuert werden können. **Steuermodul:** Das Herz des Steuermoduls ist ein Arduino Due Mikrocontroller. Die hohe Flexibilität des Systems sowie die Funktion zur automatischen Erkennung des angeschlossenen Aktuators, Selbsttests und Notausfunktionen erfordern eine Vielzahl von Steuersignalen. Die Aufgabe diese Signale zu steuern übernimmt der Mikrocontroller. Für ein Steuersignal werden ~30 mA vom entsprechenden Mikrocontroller I/O-Pin benötigt. Bei einer maximalen Strombelastung von 130 mA für alle I/O Mikrocontrollerpins könnten somit maximal 4 Steuersignale gleichzeitig aktiv sein. Dies würde bei Weitem nicht ausreichen. Zudem übersteigt die Anzahl an Steuerleitungen die Zahl der verfügbaren Mikrocontroller I/O-Pins. Die Reduzierung der erforderlichen Mikrocontrollerpins wird mittels Schieberegister durch ein Zeitmultiplexverfahren gelöst. Außerdem wird die für die Steuersignale erforderliche elektrische Leistung von einer zusätzlichen Endstufe aufgebracht. Das Steuermodul ermöglicht somit die Konfiguration des gesamten Systems.

Im Rahmen dieser Abschlussarbeit wird ein Prototyp entwickelt welcher bereits teilautomatisierte Aktuatorprüfungen vornehmen kann. Die Anforderungsanalyse des Prüfsystems und der Aktuormessungen, das Konzept des Prüfsystems, die bereits erstellten Schaltpläne für die Hardware Module des Prüfsystems sowie die vorhandenen Prototypen dienen als Basis für die Weiterentwicklung des gesamten Systems.

[1] Homepage DEWESoft, <http://www.dewesoft.com/products/sirius>

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Eigene Darstellung

Konzeption, Implementierung und Validierung einer Firewall in einer AUTOSAR Umgebung

Marc Kaiser*, Werner Zimmermann, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Mit dem Einzug von Ethernet ins Automobil spielt auch die Kommunikation mit der Außenwelt eine stärkere Rolle. Es öffnen sich hier neue Möglichkeiten, aber auch Risiken. Durch die erhöhte Anzahl an Schnittstellen zum Fahrzeug nach außen werden die elektronischen Steuergeräte (ECUs) im Auto, zu einem geeigneten Angriffsziel. In den letzten Jahren wurden von Hackern immer wieder Sicherheitslücken in Fahrzeugen, demonstriert.

Ein besonders ernstes Ereignis war für den Automobilhersteller Fiat Chrysler, die komplett ferngesteuerte Übernahme eines seiner Fahrzeuge durch „White Hat“ Hacker. Der Automobilhersteller war daraufhin gezwungen 1,4 Millionen Fahrzeuge zurückzurufen [1].

Das Netzwerk eines Fahrzeugs ist unterteilt in Bereiche, die in unterschiedlichem Ausmaß sicherheitskritisch sind (Abbildung 1).

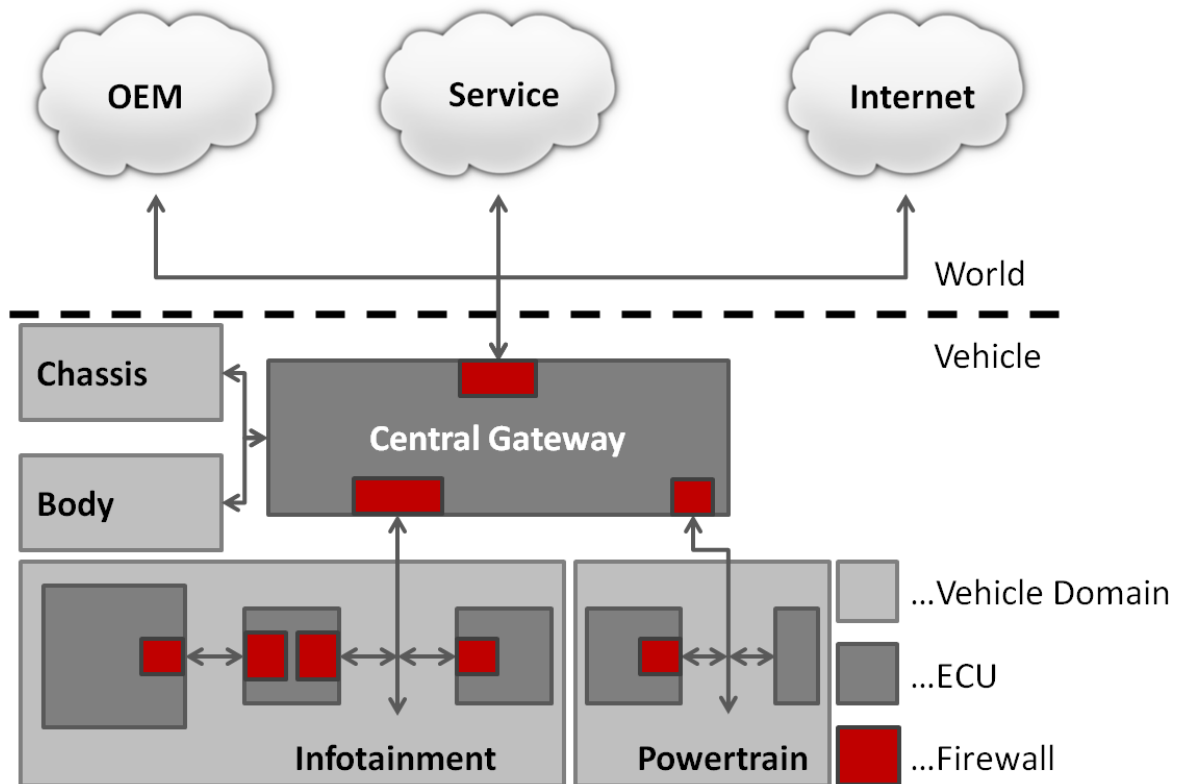


Abbildung 1: Overview Vehicle Domains

Es werden Firewalls benötigt, um unterschiedliche Sicherheitsrichtlinien in den verschiedenen Bereichen zu erzwingen. Thema dieser Arbeit ist die Konzeption, Implementierung und Validierung einer solchen Firewall für Steuergeräte. Es gibt verschiedene Ausprägungen des Firewall-Konzepts. Das am häufigsten eingesetzte Konzept nennt sich State-

ful Packet Inspection (SPI). Dabei werden die Datenpakete zusätzlich zu den direkt im Paket vorhandenen Informationen auch anhand des Verbindungskontextes überprüft.

*Diese Arbeit wurde durchgeführt bei der Firma Vector Informatik, Stuttgart

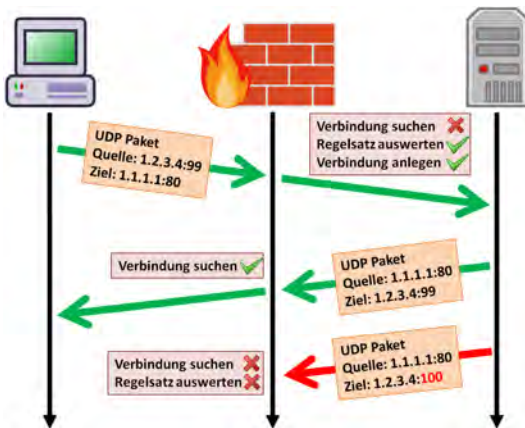


Abbildung 2: Stateful Packet Inspection

SPI Firewalls eignen sich besonders für den Embedded Automotive Sektor, da diese leistungsfähig sind und größtenteils ohne Software Updates auskommen. Updates sind nicht essenziell, da sich die Technologie nicht auf die Verteidigung gegen die sich ständig weiterentwickelnden Angriffstechniken konzentriert, sondern versucht die auf bestehenden Protokollen basierende Kommunikation soweit einzuschränken wie möglich, ohne die Anwendung zu beeinflussen. Die Leistungsfähigkeit kommt aus dem Ansatz, dass für Pakete für die eine korrespondierende Verbindung gefunden werden kann, keine komplette Evaluation des Firewall-Regelsatzes durchgeführt werden muss. Diese werden somit in einem sogenannten „schnellen Pfad“ bearbeitet, indem nur ein Plausibilitätscheck durchgeführt wird (Abbildung 2). Als Resultat dieses Ansatzes entstehen also Leistungsvorteile für Verbindungen, die lange aufrecht gehalten werden, während Leistungs Nachteile für kurze oder verbindungslose Kommunikationen entstehen.

Um einen Überblick über das Thema zu gewinnen, wurden in der Arbeit zunächst die Open Source Firewalls OpenBSD PF und netfilter/iptables auf ihr Verhalten analysiert. Die auf Basis dieser Erkenntnisse entworfene, an die speziellen Automotive-Anforderungen angepasste Firewall wurde sowohl für IPv4 als auch für IPv6 umgesetzt und umfasst die zusätzlichen Protokolle UDP, TCP, ICMPv4 und ICMPv6. Die Firewall wurde als funktionsfähiger Prototyp für die Vector Basis Software MICROSAR entwickelt. MICROSAR ist die Vector eigene Umsetzung des AUTOSAR Standards. Das Ziel von AUTOSAR ist es, die stetig wachsende Komplexität in Fahrzeugsoftware, durch eine Erhöhung der Wiederverwendbarkeit von Programmcode beherrschbar zu machen. Da-

bei stützt sich AUTOSAR vor allem auf Modularisierung und klar definierten Schnittstellen [2]. AUTOSAR Software ist unterteilt in die 3 Abstraktionsebenen, Software Component (SWC), Runtime Environment (RTE) und Basic Software (BSW).

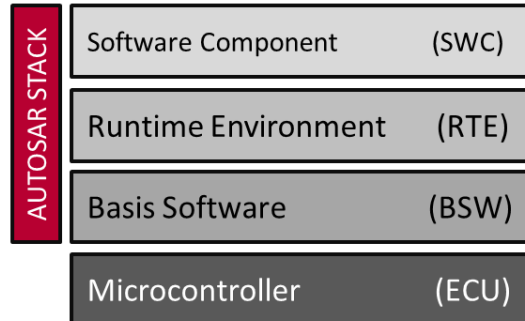


Abbildung 3: AUTOSAR Abstraktion

Hierbei setzen die SWCs die eigentliche Funktion der Software um. Das BSW stellt die, zum Teil, hardware-spezifische Grundlage dar. Das RTE ermöglicht die Kommunikation zwischen SWCs untereinander und zwischen BSW und SWC (Abbildung 3). Die Entwicklung, der Firewall wurde mithilfe des Simulationstools CANoe durchgeführt. Fragestellungen, die in der Arbeit betrachtet werden, beziehen sich auf geeignete Architekturen der Firewall und eine effiziente Struktur der Firewall Regeln und deren Auswertung. Mit Hilfe von CANoe wurden automatische Testfälle in C# entwickelt. Hierbei erfolgte eine Unterteilung in Blackbox und Whitebox Tests sowie die Erstellung verschiedener Äquivalenzklassen, um die Tests in ihrem Umfang einzuschränken. Die Whitebox Tests sind so ausgelegt, dass interne Mechanismen wie das Überprüfen der Übereinstimmungskriterien getestet werden. Die Blackbox Tests stellen die allgemeine Funktion der Firewall sicher.

Firewalls arbeiten auf Store-and-Forward Basis. Als Resultat erhöht sich die Latenz von, durch Firewalls geschützter, Kommunikation. Zum Schluss der Arbeit soll also eine Leistungsanalyse auf einem VC121 32-Bit Dual-Core PowerPC stattfinden, um Aussagen über den Speicherbedarf und die Laufzeiteigenschaften liefern zu können. Untersucht werden sollen außerdem noch Optimierungsmethoden, wie die Anwendung von Hashtabellen um die Suche nach passenden Verbindungen mit annähernd konstanter Laufzeit zu ermöglichen.

[1] <http://www.spiegel.de/auto/aktuell/nach-hackerangriff-fiat-chrysler-ruft-suvs-von-jeep-zurueck-a-1045294.html>

[2] https://vector.com/vi_autosar_de.html

Entwurf und prototypische Implementierung einer Umgebung zur massiven parallelen Simulation unter der Verwendung von GPGPU-Hardware

Daniel Klaiber*, Martin Stämpfle, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Der Geschäftsbereich Chassis Systems Control (CC)(Abbildung 1) entwickelt innovative Komponenten, Systeme und Funktionen der Fahrzeugsicherheit, -dynamik und Fahrerassistenz. Aktive Sicherheitssysteme erkennen Risikosituationen und greifen zielgerichtet in die Fahrzeugdynamik ein, um die Stabilität des Fahrzeugs wiederherzustellen. Ist ein Unfall nicht mehr zu verhindern, schützen passive Sicherheitssysteme Fahrzeuginsassen sowie Fußgänger, wie zum Beispiel Airbags und Gurtstraffer und senken das Verletzungsrisiko.[1] Das zentrale Airbag-Steuergerät erkennt aus Sensorsignalen die Stärke und die Richtung eines Aufpralls und steuert für maximalen Insassenschutz bedarfsgerecht die Rückhaltemittel im Fahrzeug bzw. den Aufprallschutz für Fußgänger.[2]

Passive Safety & Sensors	
Vision of CC-PS	Quality – Safety – Agility Safety for All "We stand for safety"
Mission	We develop, produce and sell state of the art automotive safety functions, Airbag-ECUs and Sensors. We contribute to comprehensive integrated safety systems for CC
Values	Our values are the Bosch Values
Priorities	<ul style="list-style-type: none"> • Results (OVC and Customer Results) • Growth: 25% SoM • Global Organization - local action- joint responsibility for PS- global result

Abbildung 1: CC-PS Vision

Bereits frühzeitig werden für spätere Serienfahrzeuge entsprechende Vorserienautos auf unterschiedliche Unfallarten überprüft. Bevor man den realen Versuch startet, wird er am Computer simuliert. Das nennt man virtueller Crash.[4]

Abzustimmen sind:

- Steifigkeit der Fahrgastzelle
- Entstehende Beschleunigungsabläufe
- Angepasste Insassenschutzsysteme



Abbildung 2: Prozesskette eines Airbagssystems

Ein typischer Ablauf einer Kollision (Abbildung 2):

1. Pre-Crash-Phase

Bereits die Millisekunden vor dem Crash können genutzt werden, um die Insassenschutzsysteme bestmöglich vorzubereiten. Sensor- und Auswertesysteme erfassen die Situation um das Fahrzeug und im Innenraum: z.B. Fahrzustand, Annäherung an ein Objekt, Position der Fahrzeuginsassen. Mit diesen Informationen lassen sich schon vor dem Aufprall die Auslöseschwellen der Rückhaltemittel anpassen. Bei Bedarf können schon jetzt reversible Rückhaltemittel aktiviert werden, wie beispielsweise die Gurtstraffer.

2. Crash-Phase

Die Sensoren erfassen präzise und schnell die Richtung und die Schwere des Aufpralls. Das Airbag-Steuergerät verarbeitet diese Informationen und steuert die Aktivierung der Rückhaltemittel.

3. Post-Crash-Phase

Nach dem Crash schaltet das Airbag-Steuergerät die Kraftstoffzufuhr ab, entriegelt die Türen und schaltet die Stromzufuhr ab. Ein modernes System kann automatisch einen Notruf mit der genauen Fahrzeugposition absetzen.

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Ditzingen



Abbildung 3: Oberfläche von MDSng

Da die Kosten zu hoch wären, nach jeder Änderung der Kalibrierung erneut einen wirklichen Crash zu fahren, müssen alle Crashes, die getestet werden müssen, simuliert werden. Das große Problem dabei ist, dass es große Datenmengen sind. Für jeden Crash gibt es bis zu 100.000 Iterationen, da er in der Robustheit, Amplitude, Offset und Frequenz durchvariiert wird. Bisher werden die Simulationen auf verschiedenen Cloud-PCs mit der Bosch-Software MDSng (Abbildung 3) berechnet und nimmt viel Zeit in Anspruch. Nun soll eine Umgebung erschaffen werden, in der man testet, wie lange eine mehrfache Filterung wie beispielsweise ein Tiefpass dauert im Vergleich zu den bisherigen Lösungen. Interessant ist dabei auch ob auch eine der Lösungen zu den vorhandenen Sensormodelle passt.

Da in den meisten Notebooks bei BOSCH eine NVIDIA Grafikkarte eingebaut ist, wird im Rahmen dieser Bachelorarbeit die NVIDIA Architektur CUDA verwendet. Durch diese kann die Rechenleistung eines Systems, durch den Einsatz des Grafikprozessors, deutlich gesteigert werden. Es sind schon viele CUDA-fähige Grafikprozessoren im Einsatz. Softwareprogrammierer, Wissenschaftler und Forscher verwenden CUDA in unterschiedlichen Bereichen, von der Bild- und Videoverarbeitung über Computerbiologie und -chemie, numerische Strömungssimulationen, CT-Bildrekonstruktion und seismische Analysen bis hin zu Raytracing und vielem mehr. Die CUDA® Plattform für parallele Berechnungen umfasst einige einfache C und C++ Erweiterungen, um fein- und grobkörnige Parallelität von Daten und Aufgaben auszudrücken. Der Programmierer kann die Parallelität in High-Level-Sprachen wie C, C++ oder Fortran oder in offenen Standards wie mit OpenACC Direktiven ausdrücken. (Abbildung 4) Mit Tausenden Grafikprozessorbeschleunigten Anwendungen und Tausenden veröffentlichten Forschungsberichten ist die CUDA Plattform für parallele Berechnungen solide etabliert.[3]

Abbildung 4 zeigt einen Vergleich von Standard C-Code und parallelem C-Code. Der Standard C-Code verwendet eine einfache Schleife zur Berechnung der SAXPY-Funktion. Der parallele C-Code verwendet die CUDA-Direktive `__global__` und `__cuda__` zur Parallelisierung der Berechnung über Tausende von GPU-Kernen.

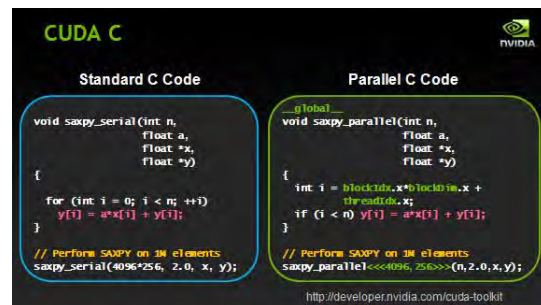


Abbildung 4: Vergleich von Standard C-Code und parallelem C-Code

[1] Inside Bosch Homepage – Chassis System Control 2015

[2] Inside Bosch Homepage – Passive Safety 2015

[3] <http://www.nvidia.de/object/cuda-parallel-computing-de.html>

[4] <http://www.volkswagen.de/de/technologie/technik-lexikon/crashtests.html>

Bildquellen:

- Abbildung 1: Robert Bosch GmbH (About Engineering Passive Safety)
- Abbildung 2: Robert Bosch GmbH (About Engineering Passive Safety)
- Abbildung 3: eigene Darstellung
- Abbildung 4: <http://www.nvidia.de/object/cuda-parallel-computing-de.html>

Erstellung einer KI für ein Grand Strategy Spiel – Anwendung und Analyse verschiedener Entwicklungsprinzipien

Simon Kuhn*, Andreas Rößler, Reinhard Schmidt

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Künstliche Intelligenz ist ein wichtiges informationstechnisches Fachgebiet, das in vielen technischen Branchen Anwendung findet. Das grundlegende Ziel ist die Simulation menschlichen Verhaltens, was beispielsweise bei der Entwicklung automatisch agierender Autos zum Einsatz kommt. Aber auch im Bereich der Unterhaltungssoftware sind KI-Implementierungen sehr häufig. Der Großteil aller Spieleprogramme setzt eine KI ein. Sie kann in diesen zahlreiche Aufgaben, wie zum Beispiel die Simulation eines Gegenspielers, übernehmen. Die Analyse und Evaluation der verschiedenen Entwicklungsprinzipien, die bei einer solchen Spiele-Entwicklung angewandt werden, ist das Thema dieser Abschlussarbeit. Zunächst wird die künstliche Intelligenz als akademisches Fachgebiet erfasst und dargestellt. Dazu werden die Anfänge und die historische Entwicklung ausführlich beschrieben. Die besprochenen Inhalte müssen jedoch auf das gewünschte Anwendungsgebiet, das der Computerspiele, eingegrenzt werden. An dieser Stelle wird beispielsweise der Unterschied zwischen einem computergesteuerten Gegenspieler und einem Non-Player Character erfasst [1]. Der aktuelle Stand der Technik in diesem Bereich wird analysiert und evaluiert. Die Betrachtung von Decision Making spielt eine wichtige Rolle, deren Ziel die Entscheidungsfindung auf Basis einer momentanen Situationsanalyse ist. Unterschiedliche Implementierungsmöglichkeiten wie State Machines, Decision Trees oder auch Behaviour Trees werden beschrieben und verglichen [2]. Im Weiteren wird der Stand der Entwicklungsarbeit bezüglich der Lernfähigkeit einer KI betrachtet. Hierbei wird der Gedanke des Decision Makings durch dynamisch erstellte Decision Trees weitergeführt. Darüber hinaus werden Methoden zur Aktionsvorhersage und zur Parameter Modifikation besprochen [2].

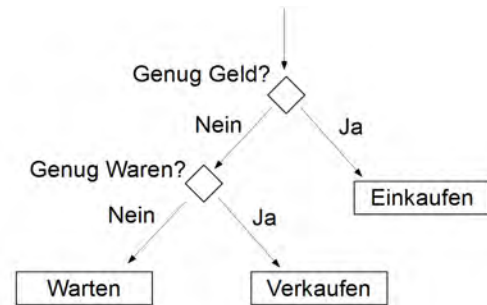


Abbildung 1: Beispielhafter Decision Tree.

Die Analyse und Evaluierung verschiedener Entwicklungsprinzipien von Spiele-KI soll in diesem Fall durch die Umsetzung eines möglichst industrienahen und damit authentischen Projekts ermöglicht werden. Aus diesem Grund wurde das Konzept für einen aufwendigen Spieleprototyp entwickelt. Dieses beinhaltet zahlreiche programmatische und grafische Features, die benötigt werden, um ein tatsächliches Spielgefühl zu vermitteln, sowie zahlreiche Interaktionsmöglichkeiten, die dem Spieler und der KI gleichermaßen zur Verfügung gestellt werden. Auch die Anforderungen an die KI müssen explizit festgelegt werden. Dabei spielt besonders die bereits durchgeführte allgemeine Analyse von künstlicher Intelligenz in Computerspielen eine wichtige Rolle. Die Rahmenbedingungen der KI als Spielfeature beinhalten beispielsweise Spielbalance oder die überzeugende Simulation eines menschlichen Gegenspielers. Auch technische Anforderungen, wie die resultierende Performance, werden betrachtet.

Basierend auf diesen Überlegungen wurde ein lauffähiges Spielprogramm entwickelt. Dieses stellt eine klassische Konfliktsituation des Strategiegenres dar, in der die virtuelle Spielwelt verschiedenen Ländern zugeteilt wird, welche um die Vorherrschaft kämpfen. Das Spiel beinhaltet drei Spielelemente: Länder, Provinzen und Armeen. Ein Land kann von einem Spieler kontrolliert werden, wodurch mit den hierarchisch tiefer liegenden Provinzen und Armeen interagiert wird. Provinzen sind eine vereinfachte Aufteilung der Spiel-

*Diese Arbeit wurde durchgeführt an der Fakultät Informationstechnik

welt. Sie sorgen für die Ressourcen-Produktion und dienen dazu, die Lage von Armeen in der Spielwelt darzustellen und zu verwalten. Armeen schließlich sind die einzigen Spielelemente die vom Nutzer mehrmals instanziiert und bewegt werden können. Sie können feindliche Provinzen erobern und müssen dementsprechend die eigenen Provinzen gegen feindliche Armeen verteidigen.



Abbildung 2: Die Übersichtskarte zeigt diverse Länder und deren Provinzen.

Wie der menschliche Spieler so kann auch die KI die Kontrolle über ein Land übernehmen. Im Normalfall werden alle Länder, die nicht vom Spieler gesteuert werden, einer KI zugeteilt. Eine KI erhält während des Spielverlaufs zyklisch Daten, die sie für ihre Entscheidungsfindung benötigt. Diese Daten repräsentieren den momentanen Spielzustand, können aber auch komplexere Parameter, wie Bewertungen vorheriger Aktionen, beinhalten, was für die Lernfähigkeit benötigt wird. Für die Verarbeitung dieses Inputs, erstellt die KI eine interne Logikstruktur, indem beispielsweise auf Basis bereits bekannter Spielzüge die Entropie der zu

testenden Attribute berechnet wird. Hat sich schließlich die KI für den nächsten Spielzug entschieden, wird dieser an das Spiel zurückgegeben und für das entsprechende Land umgesetzt.

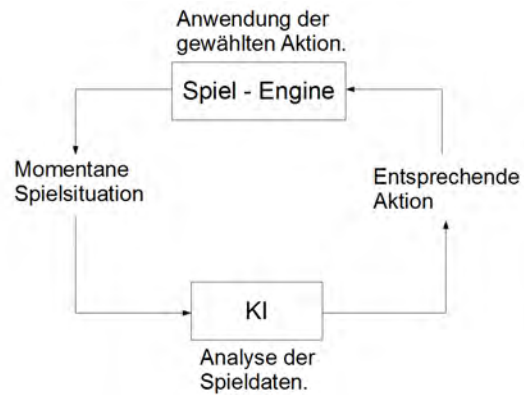


Abbildung 3: Arbeitsweise der KI im Programmverlauf.

Im letzten Kapitel der Arbeit wird die entwickelte KI, sowie deren Entwicklungsprozess analysiert und bewertet. Dazu werden die Kriterien des zuvor erstellten Konzepts erneut aufgegriffen. Wichtig für die Bewertung ist die technische und logische Leistungsfähigkeit, aber auch die Umsetzung eines menschenähnlichen Verhaltens, da dies, wie bereits erwähnt, eine der Grundanforderungen an eine KI darstellt. Nach einer abschließenden Evaluierung des Gesamtprojekts werden daraus folgende neue Herausforderungen zusammen mit ersten Lösungsansätzen formuliert. Es entsteht ein erweiterter Ausblick auf zukünftige Einsatzmöglichkeiten von KI und deren effektiver Entwicklung.

[1] Schell, J. ; The Art of Game Design – A Book of Lenses, CRC Press, 2008, 1. Auflage

[2] Millington I. / Funge J. ; Artificial Intelligence for Games, CRC Press, 2009, 2. Auflage

Bildquellen:

- Abbildung 1: Simon Kuhn – 2015
- Abbildung 2: Simon Kuhn – 2015
- Abbildung 3: Simon Kuhn – 2015

Entwicklung und Implementierung einer Anwendung für Microsoft SharePoint 2013 zur Beantragung und Genehmigung von Benutzerrechten für Microsoft Team Foundation Server unter Verwendung des Architekturmusters MVC im agilen Entwicklungsumfeld von Scrum

Tanja Last*, Andreas Rößler, Reinhard Schmidt

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Planung von Projekten ist in jeglicher Hinsicht eine komplexe Angelegenheit. Egal ob im Straßenbau, der Politik oder im Bereich der Informationstechnik versucht man die Entwicklung und Dauer eines Projektes so präzise wie möglich voraus zu bestimmen, um Fehler zu vermeiden und Kosten zu sparen. Für diese Planung und Entwicklung werden gerade im Bereich der Softwareentwicklung mächtige Management Werkzeuge, die so genannten Application-Lifecycle-Management (ALM) Tools verwendet. Diese Tools werden vor allem in Unternehmen eingesetzt, in denen eine Multiplizität an verschiedenen Projekten gleichzeitig entwickelt wird.

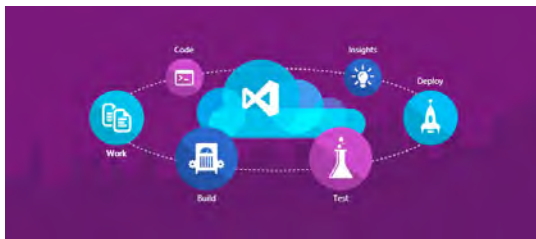


Abbildung 1: Übersicht über TFS Funktionen

Diese Bachelorarbeit entstand ebenfalls mit einem ALM Tool. Es wurde Microsoft Team Foundation Server (TFS) verwendet. Diese Software ist für die Verwaltung und Ablage von Entwicklungsprojekten zuständig und umfasst verschiedenste Komponenten. Von agilen Entwicklungsmethoden über Versionskontrolle bis hin zu Tests und Continuous Integration begleitet Microsoft Team Foundation Server den kompletten Entwicklungsprozess von Anfang bis Ende. Auch Team Foundation Server von Microsoft verwaltet die Projekte aller Teams eines einzigen Unternehmens. Durch die Ansammlung von mehreren Projekten an einer Stelle muss darauf geachtet werden, dass sich der Zugriff von unterschiedlichen Teams ausschließlich auf die Projekte begrenzt, mit denen das Team arbeitet. In TFS wird die Benutzerverwaltung über Berechtigungsgruppen

geregelt. Benutzer werden Gruppen mit speziellen Rechten zugeordnet. Beispielsweise existiert eine Administrator Gruppe, welche projektspezifische Rechte wie beispielsweise das Hinzufügen und Löschen von Team Mitgliedern für ein bestimmtes Projekt besitzt, oder eine Gruppe die nur über Lese-Rechte für ein spezielles Projekt verfügt. Jedes spezifische Projekt hat demnach seine ihm spezifisch zugeordneten Gruppen mit den jeweils zugeordneten Benutzerrechten. Diese Benutzergruppen müssen erstellt und anschließend stets beim Hinzufügen neuer Benutzer editiert werden.

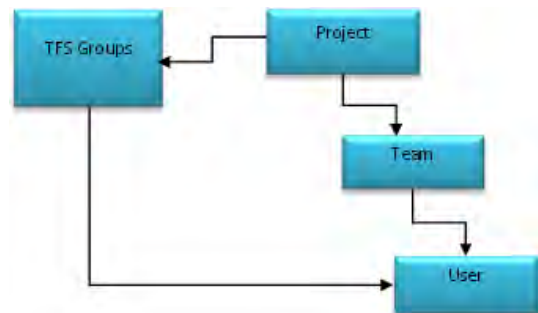


Abbildung 2: Ausschnitt aus der TFS Struktur

Dazu müssen diese neuen Benutzer einen Antrag an die Administratoren des ALM Werkzeuges stellen. Ein neuer Benutzer soll selbstständig in der Lage sein alle für die Administratoren notwendigen Daten zusammenzutragen und diese mit Hilfe eines Online-Formulares den Administratoren zukommen zu lassen. Die Daten werden anschließend in einer Datenbank zwischengespeichert. Über das gleiche Webinterface gelangt der ALM-Administrator dann zu einer Übersicht in der er die im Formular angegebenen Daten überprüfen kann und dem Antrag entweder stattgeben oder ihn abzulehnen kann. Mit Bestätigung des Antrages wird der neue Benutzer automatisch den von ihm gewünschten Gruppen hinzugefügt. Dieser Vorgang muss also nicht mehr manuell vom ALM-Administrator durchgeführt werden.

*Diese Arbeit wurde durchgeführt bei der Firma Philips Medizin Systeme, Böblingen

Damit jedem Benutzer diese Anwendung zur Verfügung steht, muss sie an einem zentralen Ort und somit für jeden zugänglich gespeichert werden. Die betreuende Abteilung im Unternehmen Philips Medizin Systeme Böblingen verwendet zum Austausch von Dokumenten und als zentralen Zugangspunkt das Produkt Microsoft SharePoint. SharePoint bietet die Möglichkeit neben Dokumenten auch so genannte Add-ins zu teilen. Add-ins sind in diesem Fall Anwendungen, die auf SharePoint gehostet werden. Statt auf SharePoint Servern, kann die Anwendung ebenfalls auf eigenen Servern als externe Webanwendung gehostet werden, steht dann aber dennoch über SharePoint zur Verfügung. Die Anwendung dieser Bachelorarbeit läuft ebenfalls als Webanwendung über SharePoint.

In der Softwareentwicklung wird außerdem großer Wert auf die Erweiterbarkeit, Wartbarkeit und Austauschfähigkeit einer Software gelegt. Es soll auch neuen Mitarbeitern möglich sein, sich schnell in den Code der Software einzuarbeiten und diesen leicht auszutauschen. Als Hilfe hierfür dienen in diesem Bereich Architektur- und Designmuster. In dieser Bachelorarbeit wurde der Schwerpunkt auf die Verwendung und Umsetzung des Architekturmusters Model-View-Controller, kurz auch MVC genannt, gelegt.

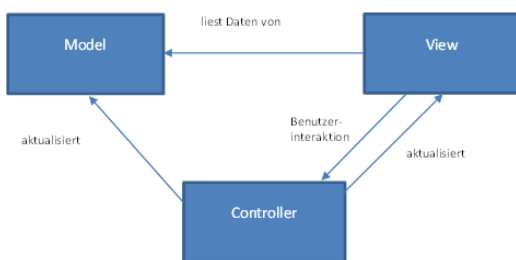


Abbildung 3: Architekturmuster MVC

Die Grundidee dieses Architekturmusters ist die Trennung von Datenhaltung, Geschäftslogik und Visualisierung. Das Model repräsentiert die Datenhaltung, der Controller ist für das Ausführen der Geschäftslogik und die View für die Visualisierung der Oberfläche zuständig.

Sowohl Model als auch Controller sind C#-Klassen, die View wird mit Hilfe von HTML, CSS und Razor strukturiert. Razor ist eine

Technologie, die es ermöglicht, C#-Code innerhalb von HTML zu programmieren. Dieser Code wird dann zur Laufzeit dynamisch zu einer HTML-View gebaut. Über Razor lassen sich HTML-Tags wie Input-Felder, Select-Menüs oder Checkboxes direkt mit den Properties des Models verbinden. Durch die unkomplizierte und klare Syntax von Razor wird der Code für die View übersichtlich und leichter lesbar. Über Razor und die Beziehung der HTML-Tags zu den Properties des Models entsteht dann die Verbindung zwischen Model und View. Für die Oberflächen-Logik wird von der Javascript Bibliothek jQuery Gebrauch gemacht. Zur Datenübertragung wird unter anderem die Technologie Ajax (Asynchronous JavaScript and XML) verwendet. Ajax wird von jQuery unterstützt und wird benutzt um asynchron Daten zwischen der View und dem Controller auszutauschen ohne bei jeder Datenübertragung die komplette Website aktualisieren zu müssen. Es werden lediglich die benötigten Daten übertragen, was zu einer Entlastung der Servers führt.

Als Motivation dieser Bachelorarbeit dienen in erster Linie die Zeitersparnis, Übersichtlichkeit, Effizienz der Arbeit und die Vereinfachung eines Vorganges. Durch die Automatisierung, verliert ein Administrator von TFS keine Zeit mehr, richtige Gruppen herauszusuchen und diesen Gruppen Benutzer hinzuzufügen. Die Vereinfachung steht in direkter Beziehung hierzu. Durch einen einfachen Klick auf einen Button ist die Genehmigung oder Ablehnung eines Antrages abgeschlossen und komplexe Vorgänge für das Suchen und Hinzufügen bleiben einem TFS Administrator erspart. Ferner werden zwei Vorgänge zu einem zusammengefügt. Die Anfrage und die Annahme oder Ablehnung der Genehmigung, kurz gesagt die Antwort der Anfrage, sind beide zentral an einem Ort zusammengefasst. Allerdings steht die Ansicht des tabellarischen Datenspeichers ausschließlich den Administratoren von Team Foundation Server zur Verfügung. Da sich alle Anfragen in einem gemeinsamen Datenspeicher befinden und diese in einer tabellarischen Übersicht innerhalb der Webanwendung eingesehen werden können, dient die erstellte Software somit als auch Nachschlagewerk.

Bildquellen:

- Abbildung 1: <http://www.wintellect.com/devcenter/wp-content/uploads/2015/03/microsoft-visual-studio-online.jpg> Zugriff am 30.11.2015
- Abbildung 2+3: eigene Darstellung

Entwicklung eines XCP-Slaves zum Messen und Parametrieren eines Universalsteuergerätes

Christoph Lenger*, Hermann Kull, Manfred Dausmann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In heutigen Fahrzeugen werden immer mehr Steuergeräte verbaut, um die große Menge an Informationen zu verarbeiten, welche die verschiedenen Sensoren liefern. Innerhalb des Steuergerätes läuft eine Software, von der die Sensordaten verarbeitet und die Ergebnisse der einzelnen Berechnungen an die Aktoren weitergegeben werden. Diese Software ist meist sehr komplex und enthält sowohl Regelungen als auch Algorithmen. Um die einzelnen Parameter einer Regelung passend zu bestimmen, werden viele Stunden am PC mit einer gängigen Simulationssoftware verbracht. Wird die Software danach im Steuergerät ausgeführt, kann es zu Situationen kommen, in denen das (Fehl-) Verhalten nur schwer gedeutet werden kann. Dann heißt es, so schnell wie möglich die Ursache zu finden! Für den Applikationsentwickler ist es in solchen Situationen von großem Vorteil, wenn er die Möglichkeit hat, in das Innere des Steuergerätes zu blicken, genauer gesagt, wie sich die einzelnen Signale während eines Berechnungszyklus verhalten. Im normalen Anwendungsfall ist ein Steuergerät eine sogenannte „Blackbox“, das heißt, es lassen sich nur der Input und der Output des Steuergerätes erfassen, allerdings keinerlei Details über den internen Ablauf. Mit Hilfe des XCP-Protokolls ist jedoch genau dies realisierbar. Es ermöglicht, die Parameterwerte eines Steuergerätes während der Laufzeit zu verstellen und gleichzeitig Messsignale zu erfassen [1]. XCP steht für „Universal Measurement and Calibration Protocol“ und ist ein Bus-unabhängiges Master-Slave Kommunikationsprotokoll, um Steuergeräte mit Kalibriersystemen zu verbinden. Dieses Protokoll wurde vom Verein ASAM e.V. (Association for Standardisation of Automation and Measuring Systems) standardisiert. Es besteht im Allgemeinen aus zwei voneinander unabhängigen Schichten: der Protokoll- und der Transportschicht [2].

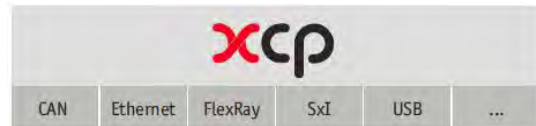


Abbildung 1: Aufteilung des XCP-Protokolls in Protokoll- und Transportschicht

Wie in Abbildung 1 dargestellt, können verschiedene physikalische Transportschichten realisiert werden. Ziel dieser Bachelorarbeit ist es, einen XCP-Slave nach der aktuellen Spezifikation (V1.3.0) zu entwickeln, um das Messen und Parametrieren eines Universalsteuergerätes zu ermöglichen. Als Transportschicht wird in diesem Fall CAN verwendet und als Hardware ein Universalsteuergerät der GIGATRONIK Stuttgart GmbH.



Abbildung 2: GIGABOX gate XL

Eine XCP Message besteht aus einem Header, dem eigentlichen XCP Packet und einem Tail (siehe Abbildung 3). Header und Tail können, je nach verwendeter Transportschicht, unterschiedlich aufgebaut sein. Das XCP Packet besteht immer aus den folgenden drei Teilen: dem Identification-, Timestamp- und dem Data-Field.

*Diese Arbeit wurde durchgeführt bei der Firma GIGATRONIK Stuttgart GmbH, Stuttgart

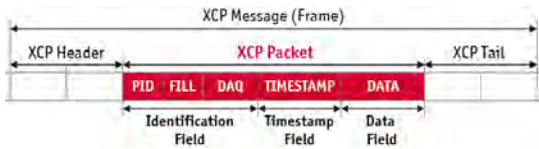


Abbildung 3: XCP-Paket

Die Kommunikation zwischen Master (Applikationssystem) und Slave (Steuergerät) ist in zwei verschiedene Arten aufgeteilt. Mit CTO (Command Transfer Object) Paketen schickt der Master die Befehle an den Slave. Jeder Befehl wird vom Slave ebenfalls mit einem CTO beantwortet. Eine solche Antwort fällt entweder positiv (RES) aus, sofern der Slave den Befehl umsetzen konnte, oder negativ (ERR). Bei einer negativen Antwort wird ein Fehlercode mitgesendet, damit der Master identifizieren kann, was genau nicht funktioniert hat.

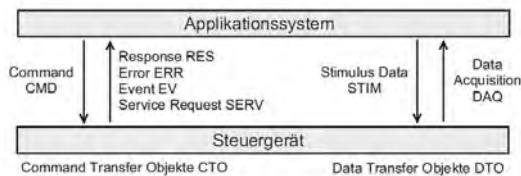


Abbildung 4: Unterteilung CTO - DTO

Die DTO (Data Transfer Object) Pakete dienen dem zyklischen Austausch von Daten zwischen Master und Slave. Je nach Verbindungsrichtung handelt es sich um Messdaten (Slave →

Master) oder Stimulus-Daten (Master → Slave). Bei der Anwendung des Protokolls werden zwei verschiedene Messverfahren unterschieden. Mit Polling kann der Master über CTOs zu einem beliebigen Zeitpunkt bestimmte Messsignale anfragen. Dieses Verfahren hat allerdings zwei bedeutende Nachteile. Zum einen antwortet der Slave auf jeden Befehl des Masters mit einem CTO und zum anderen ist es nicht garantiert, dass die Antworten des Slaves aus dem selben Berechnungszyklus stammen, wenn der Master mehrere Signale pollt. Durch die Verwendung von CTOs entstehen mehr Nachrichten auf dem Bus, welche diesen zusätzlich auslasten. Um die Auslastung zu reduzieren, gibt es das zweite Verfahren, sogenannte DAQ-Listen. Diese Listen enthalten die Messsignale die der Master zu einem bestimmten Zeitpunkt haben will. Der Transfer beruht auf DTOs, die der Slave selbständig versendet. Damit verringert sich die Auslastung des Busses, da der Master die Daten nicht einzeln anfragen muss. Zudem werden die Listen so konfiguriert, dass sichergestellt wird, dass alle darin enthaltenen Signale aus dem selben Berechnungszyklus des Slaves stammen. Somit löst dieses Verfahren gleich beide Nachteile des Polling. Das XCP-Protokoll trägt damit wesentlich dazu bei, die Entwicklung der Software von Steuergeräten zu erleichtern und das Verhalten bei der Ausführung im Steuergerät zu verstehen.

- [1] XCP–Das Standardprotokoll für die Steuergeräte–Entwicklung, Andreas Patzer & Rainer Zaiser, Vector Informatik GmbH, 2.Auflage, Stand April 2014
 [2] ASAM MCD–1 XCP, <https://wiki.asam.net/display/STANDARDS/ASAM+MCD-1+XCP>, Zugriff 02.12.2015

Bildquellen:

- Abbildung 1: XCP–Das Standardprotokoll für die Steuergeräte–Entwicklung, Andreas Patzer & Rainer Zaiser, Vector Informatik GmbH, 2. Auflage, Stand April 2014
- Abbildung 2: GIGATRONIK Stuttgart GmbH
- Abbildung 3: XCP–Das Standardprotokoll für die Steuergeräte–Entwicklung, Andreas Patzer & Rainer Zaiser, Vector Informatik GmbH, 2. Auflage, Stand April 2014
- Abbildung 4: Bussysteme in der Fahrzeugtechnik, Werner Zimmermann & Ralf Schmidgall, 5.Auflage, Springer Vieweg

Sauberes Fahren – Teilsystem Simulation

Wojciech Lesnianski*, Harald Melcher, Joachim Goll

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Um die Feinstaubbelastung durch den Straßenverkehr zu senken, ist es notwendig u.a. den Kraftstoffverbrauch der Fahrzeuge zu verringern. Dem Fahrer soll geholfen werden, seinen Fahrstil so zu verändern, dass er weniger Kraftstoff verbraucht. Die Verwendung schadstoffreicher Fahrzeuge soll erheblich eingeschränkt werden. Dazu würde aber eine statistische Erhebung über den Schadstoffausstoß der Fahrzeuge benötigt werden.

Aus diesem Grund soll ein System erstellt werden, welches den durchschnittlichen Kraftstoffverbrauch von verschiedenen Fahrzeugtypen misst, da sich der Schadstoffausstoß proportional zum Kraftstoffverbrauch verhält.

Im Projekt „Sauberes Fahren“ (SF), welches in einer Gruppen-Bachelorarbeit bei der Firma IT-Designers erstellt wird, soll mithilfe der OBD-Daten der durchschnittliche Schadstoff-

ausstoß einer Autofahrt erhoben, ausgewertet und zusammen mit den Fahrten anderer Fahrzeuge in einer zentralen Leitstelle statistisch analysiert werden.

Des Weiteren soll ein Fahrzeugsteuergerät dem Fahrer den aktuellen Schadstoffausstoß anzeigen, um durch visuelle Eindrücke zu einem umweltfreundlichen Fahren zu ermuntern. Zusätzlich soll der Fahrer auf dem besagten Gerät einen durchschnittlichen Wert, errechnet aus den Werten von Fahrzeugen gleichen Typs, darstellen können.

Das Projekt setzt sich aus 4 Teilsystemen zusammen, welche jeweils von einem Bacheloranden umgesetzt werden. Durch sorgfältig und detailliert beschriebene Schnittstellen der einzelnen Teilsysteme, ist eine Einzelarbeit der Studierenden trotz der Gruppen-Bachelorarbeit problemlos gewährleistet.

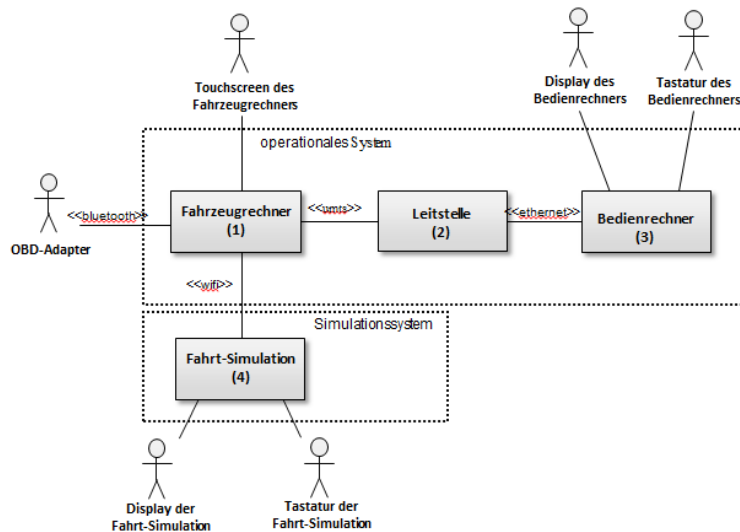


Abbildung 1: Systemarchitektur des Gesamtsystems auf der obersten Ebene

Teilsystem Simulation

Unter dem Begriff „Simulation“ versteht man den gesamten Prozess der Vorausberechnung oder des Nachstellens eines bestimmten Szenarios. Sie ist so gesehen ein Werkzeug, das bei der Analyse eines Systems helfen kann, indem sie „virtuelle Experimente“ durchführen lässt. Das wichtigste Element der Simulation ist das Modell, welches als Teil der Simulation zu verstehen ist. Es beschreibt das oft ver-

einfachte Abbild des realen Systems in der Simulation. Je höher der Detailgrad des Modells, desto näher an der Wirklichkeit ist die Simulation. Aus ihr lassen sich realitätsnähere und besser verwertbare Ergebnisse gewinnen, doch auch die Komplexität und somit die Rechenzeit steigt.[1]

Das Ziel des Teilsystems Simulation im Projekt „Sauberes Fahren“ ist es, eine Autofahrt zu simulieren und dem Teilsystem Fahrzeugrech-

*Diese Arbeit wurde durchgeführt bei der Firma IT-Designers GmbH, Esslingen am Neckar

ner die Möglichkeit zu bieten, simulierte Werte für die Geschwindigkeit und den Schadstoffausstoß abzurufen. Für diesen Zweck, und mit dem Gedanken möglichst repräsentative Daten zu erzeugen, wird für die Verkehrssimulation die Simulation of Urban Mobility (SUMO)-Suite verwendet. Diese Suite bietet eine Anzahl von fertigen Modellen, beispielsweise für den Verkehrsfluss, das Fahrverhalten oder die Fahrzeuggeschwindigkeit. Zudem hat man die Möglichkeit „echte“ Karten in die Simulation einzubinden. Allerdings geschieht das Ganze auf der Ebene der Kommandozeile und die Er-

gebnisse liegen im XML Format vor.

Das Teilsystem Simulation bietet eine in Windows Presentation Foundation (WPF) erstellte, grafische Benutzeroberfläche, welche das Hinzufügen neuer Karten, die Verwaltung und Wiederverwendung von Simulationsergebnissen, die Erstellung neuer Simulationen mit bestimmten Parametern und das Anbieten der Simulierten Werte über einen Webserver vereint und möglichst einfach macht. Durch eine Model-View-ViewModel (MVVM) Architektur unter Verwendung von Catel bleibt der Code gut und übersichtlich strukturiert.

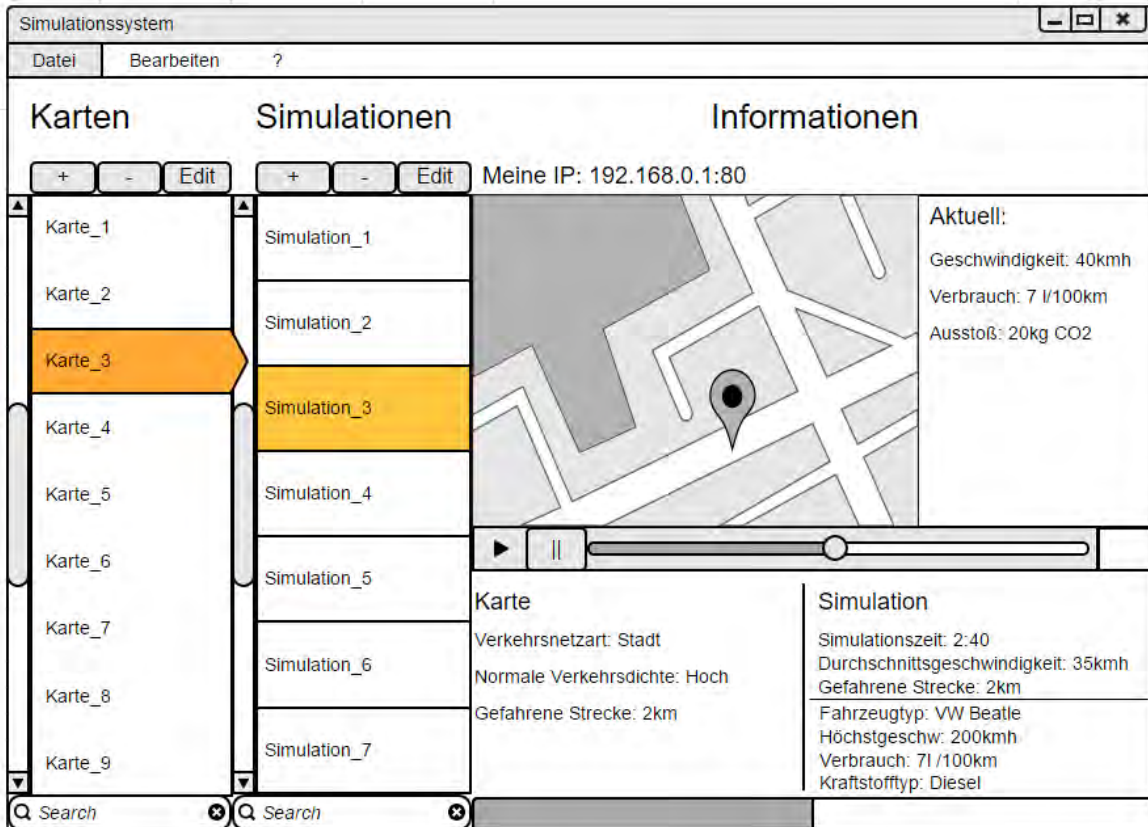


Abbildung 2: Entwurf der Benutzeroberfläche

Es ist für den Benutzer möglich eine Karte aus OpenStreetMap runterzuladen und sie in das Simulationssystem zu laden. Hierbei werden sowohl alle nötigen XML Dateien als auch Vorschaubilder automatisch erstellt. Danach kann ein Benutzer für die hinzugefügte Karte eine Simulation konfigurieren und durchführen. Durchgeführte Simulationen werden lokal persistent gespeichert und können belie-

big oft über einen REST Service dem Fahrzeugrechner zum Auslesen angeboten werden. Die Ergebnisse von Simulationen liegen im XML Format vor und beinhalten eine große Anzahl von Werten, vom Schadstoffausstoß über Verbrauch und Geschwindigkeit bis hin zur Fahrzeuglautstärke, in Abhängigkeit von der Zeit.

[1] Bungartz H.-J.; Zimmer, S.; Buchholz, M.; Pflüger, D.; Modellbildung und Simulation. s.l. : Springer, 2009.

Bildquellen:

- Abbildung 1: Eigenes Diagramm
- Abbildung 2: Eigener Entwurf

Analyse und Vergleich verteilter Datenbanksysteme und Entwurf einer verteilten Online-Shop Datenbank.

Robert Lotter*, Andreas Rößler, Reinhard Schmidt

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Datenbanken bilden die Grundlage für so gut wie jede größere Computer-Anwendung. Darum ist es natürlich von Vorteil, wenn sich die Datenbank wie gewünscht verhält. Dabei geht es vor allem um die Verfügbarkeit bzw. Ausfallsicherheit und um die Konsistenz der Daten, aber auch um die Skalierbarkeit des Datenbestandes. Alle Datenbanksysteme unterscheiden sich in diversen Aspekten und auch in ihren Konfigurationsmöglichkeiten und haben somit unterschiedliche Eigenschaften. Der markanteste Unterschied ist dabei, ob es sich um eine relationale Datenbank handelt, die ihre Daten in Tabellen mit Spalten und Zeilen speichert und sich vor allem durch Konsistenz

auszeichnet, oder um eine sogenannte NoSQL Datenbank, welche die Daten mithilfe von Schlüssel-Wert Paaren speichert und sich dadurch sehr einfach auf verschiedene Rechner verteilen lässt. Die Verteilung ist auch das Thema dieser Arbeit. Die zwei zentralen Aspekte der Verteilung sind die Fragmentierung und die Replikation, die das Ziel verfolgen, höchste Verfügbarkeit zu gewährleisten und eine sehr einfache Skalierbarkeit für das System bereitzustellen. Bei der Fragmentierung geht es darum, den Datenbestand in mehrere (am besten gleichgroße) Stücke zu zerlegen und auf verschiedene Rechner zu verteilen (siehe Abbildung 1).

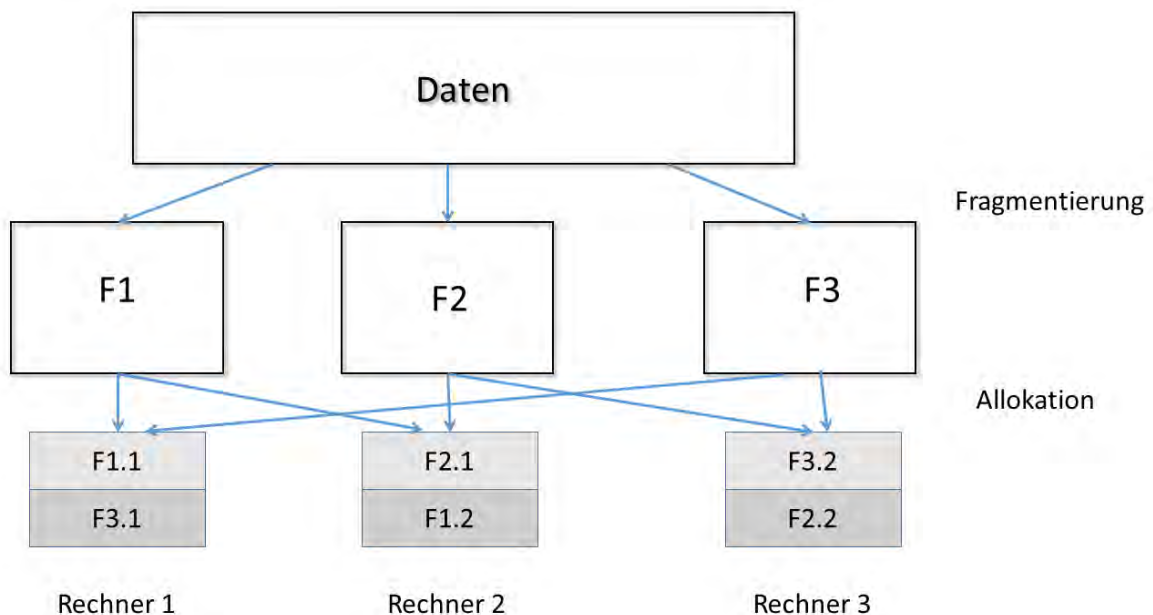


Abbildung 1: Fragmentierung und Verteilung des Datenbestandes

Um dies zu erreichen, gibt es grundsätzlich zwei Möglichkeiten. Bei der horizontalen Fragmentierung werden die Daten zeilenweise getrennt, wobei es dafür verschiedene Vorgehensweisen gibt, die jeweils unterschiedliche Vor- und Nachteile mit sich bringen. Bei der vertikalen Partitionierung werden die Daten spaltenweise getrennt, somit bleiben alle Werte eines Attributes zusammen. Diese Art

der Fragmentierung kann allerdings nicht für NoSQL Datenbanken angewandt werden, da es hier keine Spalten gibt. Bei der Replikation geht es darum, die Daten redundant auf mehreren Rechnern zu speichern. Dabei gibt es verschiedene Vorgehensweisen, die wieder diverse Vor- und Nachteile mit sich bringen (siehe Abbildung 2).

*Diese Arbeit wurde durchgeführt an der Fakultät Informationstechnik

Synchron (sofort)	+ Konsistenz gewährleistet - Längste Antwortzeit - Nur Lesezugriffe auf Slaves	+ Konsistenz gewährleistet + Schreibzugriffe auf allen Knoten ausführbar - Lange Antwortzeit - Updates können sich gegenseitig behindern
	+ Gute Performance - Inkonsistenzen - Lokale Kopien sind veraltet und können nur gelesen werden	+ Beste Performance - Inkonsistenzen - Updates können verloren gehen
Asynchron (verzögert)	Master Kopie	Überall

Abbildung 2: Replikationsstrategien

Das gängigste Modell ist die Master-Slave-Replikation [1] (siehe Abbildung 3), bei der Schreibvorgänge nur am Master-Knoten durchgeführt werden können, Lesezugriffe jedoch überall, was einen enormen Performancegewinn ergibt.

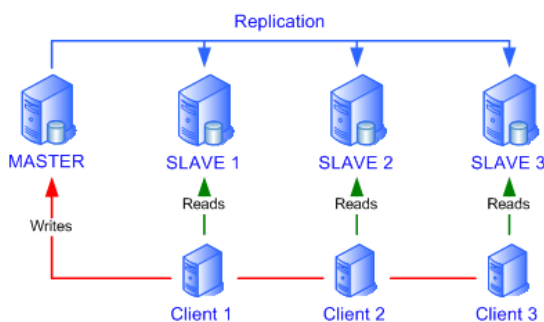


Abbildung 3: Master-Slave Replikation

Ein Ziel dieser Arbeit ist es, mehrere Datenbanksysteme zu untersuchen und zu

vergleichen. Dabei soll besonders auf die Replikations- und Fragmentierungsstrategien eingegangen werden. Daneben geht es aber auch um Aspekte wie Transaktionskonzepte, wo das Locking eine große Rolle spielt, Indexing, Datenzugriff etc. Bei den zu untersuchenden Datenbanksystemen handelt es sich um MySQL, Redis, MongoDB und Elasticsearch, welche alle als Open-Source Version zur Verfügung stehen. MySQL ist eine relationale Datenbank, Redis ist ein Key-Value Store, MongoDB ist ein Document Store und bei Elasticsearch handelt es sich um eine Suchmaschine. Aufgrund der gewonnenen Erkenntnisse soll anschließend eine Datenbankumgebung für einen Online-Shop entworfen werden. Dabei geht es von der Auswahl des Datenbanksystems, über die Konzeptionierung der Datenbank bis hin zum Entwurf und der Konfiguration eines Clusters zur Verteilung der Daten.

[1] Rahm, Erhard, Saake, Gunter und Sattler, Kai- Uwe (2015): Verteiltes und Paralleles Datenmanagement; Von verteilten Datenbanken zu Big Data und Cloud, Springer Verlag, Berlin, Heidelberg, S. 292

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Eigene Darstellung
- Abbildung 3: <http://www.netexpertise.eu/images/Replication.png>

Entwicklung und Validierung eines adaptiv-markierenden Lichts zur selektiven Beleuchtung potentieller Gefahrensituationen

Patrick Lutz*, Hermann Kull, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Über 96% der verkehrsrelevanten Informationen verarbeitet der Mensch visuell [1]. Bereits bei einem Menschen mit normaler Sehleistung reduziert sich die Sehleistung nachts auf lediglich 5% des Tagesniveaus. Zu erwähnen ist hierbei, dass die Sehleistung im gehobenen Alter noch stärker reduziert wird. Die Folgen sind, dass sich nachts (in Relation zum Verkehrsaufkommen) mehr als doppelt so viele Verkehrsunfälle mit Personenschäden ereignen als tagsüber. In den letzten 20 Jahren konnte diese Statistik – durch die Einführung von modernerem Licht und intelligenten Lichtassistentzfunktionen (wie dynamisches Kurvenlicht) – signifikant verbessert werden. Inzwischen gibt es eine Vielzahl von Lichtassistentzsystemen, welche dem Fahrer nicht nur den Fahrkorridor perfekt ausleuchten, sondern auch aktiv die Blendung des Gegenverkehrs minimieren [2].

Durch den technischen Fortschritt werden Scheinwerfer immer mehr zu wichtigen Elementen im Bereich der Verkehrssicherheit bei Nacht. Der nächste Schritt in dieser Entwick-

lung, ist der „Pixel-Scheinwerfer“, welcher eine Weiterentwicklung der bereits in Serie verfügbaren Matrix Scheinwerfer darstellt. In diesen werden mehrere einzeln ansteuerbare LED Segmente genutzt um die Fahrbahn dynamisch auszuleuchten. Der deutlich höher auflösendere Pixel-Scheinwerfer wird derzeit über verschiedene Ansätze realisiert. Ein Ansatz ist es, dies mit einem mikroskopischen LED-Array (wobei das Prinzip des Matrixscheinwerfers beibehalten wird) zu realisieren. Jedoch besteht neben dem begrenzten Bauraum – vor allem das Problem, dass die enorme Abwärme schlecht abgeführt werden kann. Ein gänzlich neues Prinzip macht sich hingegen die geringe Diffusion eines monochromatischen Laserstrahls zu Nutze, mithilfe einer aus einer Vielzahl von mikroskopischen Teilelementen bestehende Spiegelfläche wird dieser durch eine Linse gestreut und auf die Straße umgelenkt. Jeder einzelne Spiegel lässt sich dabei zwischen zwei Endpositionen umschalten, wodurch ein annähernd stufenlos dimmbares Licht vor dem Fahrzeug erzeugt werden kann.

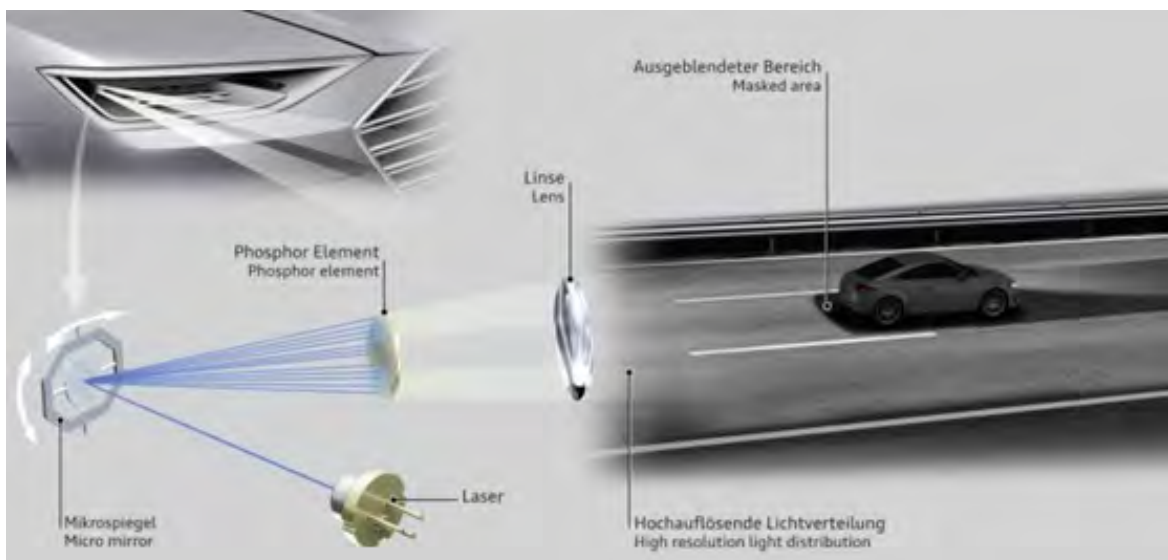


Abbildung 1: Audi Matrix Laser Technologie

*Diese Arbeit wurde durchgeführt bei der Firma IAV GmbH, Ingolstadt

Mit diesem ist es nicht nur wie bisher möglich den Gegenverkehr auszublenden, sondern Objekte oder Gefahrenstellen gezielt zu beleuchten und sogar deren Kontrast unter Zuhilfenahme von Lichtmuster oder Animationen für das Auge des Fahrers künstlich zu erhöhen. Im Vergleich zu herkömmlichen Beleuchtungen können somit Objekte deutlich früher wahrgenommen, und sogar die Aufmerksamkeit des Fahrers gezielt gelenkt werden. Ein weiterer Vorteil besteht darin, dass der Fahrer intuitiv über den Grad der Gefahr informiert werden kann, wodurch unerfahrene oder abgelenkte Fahrer unterstützt werden können. Zudem ist es vor allem auch im Angesicht des immer wichtiger werdenden Themas „Car to X“ möglich, mit einem solchen Scheinwerfersystem eine einfache auf piktografischen Projektionen basierende, Art der Kommunikation mit anderen Verkehrsteilnehmern zu erzeugen. Im Rahmen dieser Arbeit wird ein prototypischer Demonstrator aufgebaut, mit dem auf Basis von Simulationsdaten verschiedene Beleuchtungs- bzw. Markierungskonzepte visualisiert und bewertet werden können.

Zu diesem Zweck wird unter Verwendung eines Simulationstools eine Szene mit den gewünschten Parametern erstellt. Diese Art der Sensordatengenerierung ist inzwischen weit verbreitet und bietet den enormen Vorteil, dass jede beliebige Szene in kürzester Zeit generiert sowie visualisiert werden kann und komplett reproduzierbar ist. In Abbildung 2 ist beispielhaft die Szene dargestellt, die zur Erstellung der verschiedenen Gefahrenklassen (Abbildung 3) verwendet wurde.

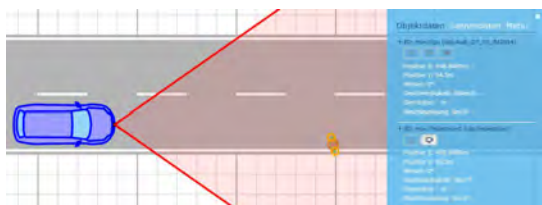


Abbildung 2: Simulierte 2D-Szene

Diese virtuellen Sensordaten wurden anschließend genutzt um mithilfe eines entwickelten Algorithmus aus der zweidimensionalen Szene, eine dreidimensionale Lichtver-

teilung zu generieren. Zunächst wurden die Klassifikationen (Typ) und Positionsdaten der Objekte erfasst. Anhand dieser und weiterer Faktoren werden dem jeweiligen Objekt eine Gefahrenklasse und eine objektspezifische Markierungsart zugewiesen. Unter Zuhilfenahme verschiedener aus der Photogrammetrie stammender Umrechnungen werden die perspektivische Position sowie Größe auf einer Leinwand berechnet, welche als Schnittebene fungiert. Anhand dieser Daten lässt sich eine Matrix generieren, welche schlussendlich die beispielhafte dreidimensionale Lichtverteilung eines zukünftigen Scheinwerfers darstellt.

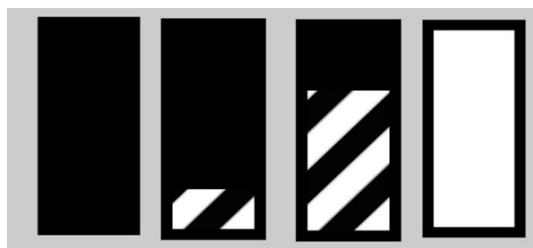


Abbildung 3: Variierende Markierungsarten
Von Links: Keine Gefahr, potenzielle Gefahr, erhöhte Gefahr, Auf Kollisionskurs (blinkend)

Das Ziel der Arbeit ist die Untersuchung und Konzeption von möglichen zukünftigen Markierungsstrategien für eine verbesserte Wahrnehmung der gegenwärtigen Verkehrssituation seitens des Fahrers. Anhand der Objektgruppe des Fußgängers wurde beispielhaft ein Algorithmus entwickelt, welcher unter der Berücksichtigung der menschlichen Wahrnehmung, Fußgänger in verschiedenen Szenen adaptiv markiert. Das Ergebnis wurde anschließend in einer modernen Halle zur Untersuchung von Licht und Lichtassistenzsystemen, mithilfe eines lichtstarken Beamers bewertet. Zukünftig ist zu erwarten, dass die Auflösung von Pixel-Scheinwerfern der Auflösung der derzeit erhältlichen Beamer nahe kommen. Deshalb darf davon ausgegangen werden, dass die Untersuchungen am Demonstrator (zumindest im Nahbereich) äquivalente Schlüsse erlauben.

[1] 9th International Symposium on Automotive Lighting – ISAL 2011 – Proceedings of the Conference • Volume 14

[2] BAST Report, Das Unfallgeschehen bei Nacht (M172), 2005

Bildquellen:

- Abbildung 1: Audi Media Info Paper – „Audi_Matrix Laser Technologie“
- Abbildung 2: Eigene Darstellung
- Abbildung 3: Eigene Darstellung

Konzipierung und prototypische Realisierung einer über magnetische Referenzpunkte geführten Navigation für ein fahrerloses Transportsystem

Patrick Mack*, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Um im Markt der fahrerlosen Intralogistiksysteme konkurrenzfähig bleiben zu können, ist das prozesssichere Arbeiten in vielen Arbeitsumgebungen notwendig. Es muss insbesondere sichergestellt werden, dass bei unterschiedlichen Bedingungen unabhängig navigiert werden kann. Die optimale Navigationsmethode muss daher für jede Umgebung unabhängig abgeschätzt werden. Einflussfaktoren für die Abschätzungen sind beispielsweise die Flexibilität bei nachträglichen Änderungen des Fahrkurses, die Kosten für die Inbetriebnahme und die Wartung.

Das bisherige fahrerlose Transportfahrzeug von BeeWaTec konnte an einer auf dem Boden aufgetragenen optischen oder an einer Magnetspur verfahren. Ein Beispielfahrzeug ist in Abbildung 1 dargestellt.



Abbildung 1: Fahrerloses Transportsystem ("Spurmeise")

In einer Umgebung, in der starker Verkehr herrscht, wird auch die optische Spur stark abgenutzt. Die Anzahl der Spurverluste über die Zeit steigt dadurch stark an. Um dieses Fehlverhalten sicher beheben zu können, musste eine berührungslose, staub- und schmutzsichere Lösung entwickelt werden. Eine Navigation anhand von Magnetreferenzpunkten deckt diese Faktoren ab. Desweiteren ist diese Navigation der nächste Schritt zu einer komplett freien Navigation.

In Zukunft sollte ein unbemanntes Fahrzeug nicht mehr einer virtuellen Spur nachfahren. Vielmehr sollten autonome Entscheidungen getroffen und der Weg eigenständig geplant werden. Dieses Ziel kann nur in definierten Einzelschritten erreicht werden. Die Implementierung einer Magnetpunktnavigation stellt einen solchen Einzelschritt dar.

Im Rahmen der Bachelorarbeit wurde das Angebot an Navigationsmethoden um eine freie Navigation, der magnetpunktgeführten Navigation, erweitert. Die bestehenden Komponenten der Software sollten, soweit möglich, unverändert bleiben. Ebenso sollte ein Kombinationsbetrieb ermöglicht werden. Bei der Umsetzung des Systems mussten diese Anforderungen berücksichtigt werden.

Neben dem Konzept und der Implementierung der zusätzlichen Module mussten auch Magnetpunkte auf die Nutzbarkeit mit dem verwendeten Sensor getestet werden. Der Messbereich des eingesetzten Magnetsensors lag hierbei außerhalb des vom Hersteller spezifizierten Bereichs. Dies machte eine Validierung des Sensors notwendig. Während dieser Validierung wurde auch die Wahl der magnetischen Referenzpunkte getroffen. Der Fokus bei der Validierung lag auf der Verfahrengenauigkeit und der Wiederholgenauigkeit. Die Länge der Sensorleiste ist dabei optimalerweise so lang wie das Fahrzeug breit ist, siehe Abbildung 3.

*Diese Arbeit wurde durchgeführt bei der Firma BeeWaTec-Systems, Pfullingen

Durch den Kombinationsbetrieb sollte es möglich sein, anhand von Positionen eines Fahrkurses präzise und sicher durch enge Passagen zu navigieren. Ein weiterer Vorteil, der sich durch den Kombinationsbetrieb ergibt, ist die Abdeckung der Nachteile der einen Navigationsmethode durch die Vorteile einer anderen.

Um die Schnittstellen mit dem bestehenden System optimal zu definieren, wurde das System im ersten Teil der Bachelorarbeit detailliert analysiert. Dadurch konnte zudem der Umfang des benötigten Quellcodes reduziert werden.

In der Magnetpunktnavigation werden die vorhandenen Magnetpunkte zur Korrektur der ansonsten virtuellen Fahrt verwendet. Dadurch ergibt sich eine Randbedingung für den Abstand zweier Magnetpunkte: Der Abstand der Punkte darf maximal so groß sein, dass das Fahrzeug unter Berücksichtigung aller Bewegungs- und Messungenauigkeiten den nächsten Referenzpunkt zuverlässig erreichen kann.

Die Abweichung des Fahrzeugs zu der virtuellen Spur wird beim Erreichen eines Magnetpunktes berechnet. Der Fahrkurs einschließlich der benötigten Korrekturen wird dann erneut berechnet. Ein Bahnregler führt die berechneten Korrekturen aus. Die korrigierte Bahn wird so lange gefahren, bis ein weiterer Referenzpunkt erkannt wird und erneut

Kurskorrekturen berechnet sind. Dieser Ablauf ist in Abbildung 2 dargestellt.

Um die Entwicklungsgeschwindigkeit zu erhöhen und um schneller Testergebnisse zu erhalten, wurden die beiden Fahrmöglichkeiten „Gerade“ und „Kurve“ schrittweise implementiert.

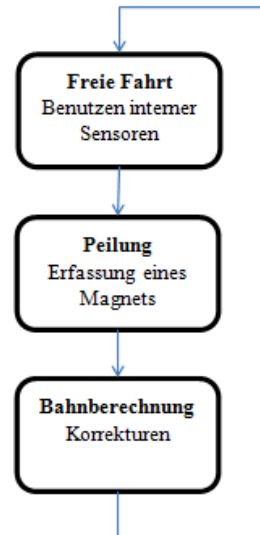


Abbildung 2: Ablauf freie Fahrt

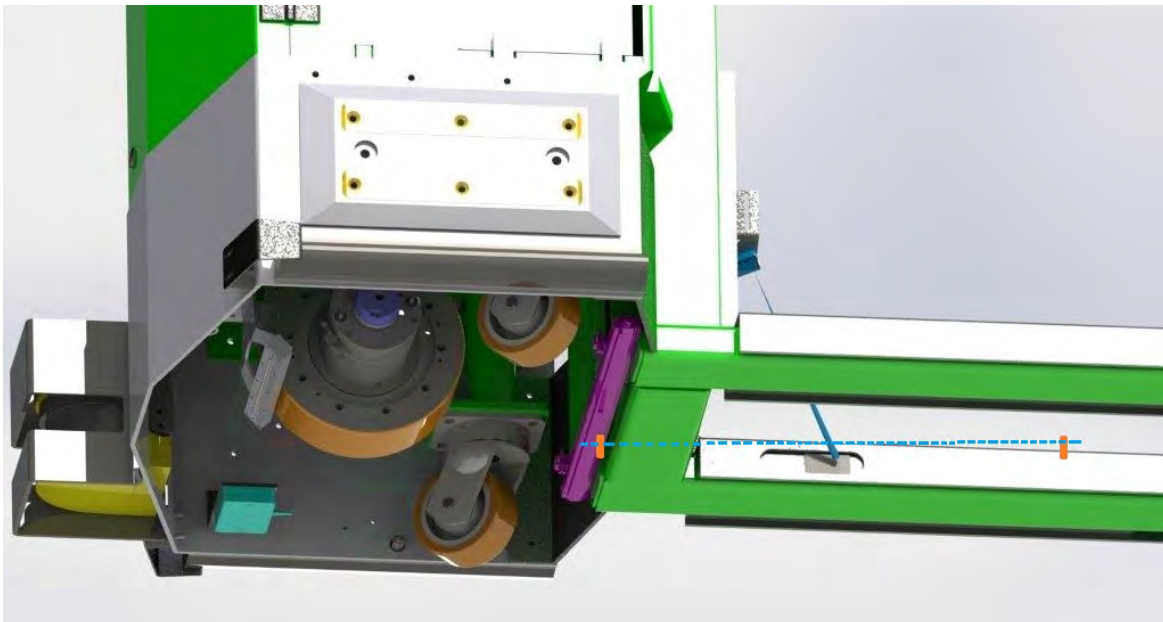


Abbildung 3: Erkennung der Referenzpunkte (Orange) und Verbindung durch virtuelle Gerade (Blau)

Bildquellen:

- Abbildung 1, 3: BeeWaTec-Systeme
- Abbildung 2: Eigene Erstellung

Konzeption und Realisierung eines Fahrzeugrechners als Teil eines verteilten Systems zur Erfassung und Auswertung der Emissionswerte von Fahrzeugen

Daniel Majorow*, Harald Melcher, Joachim Goll

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Das beliebteste und am meisten verbreitete Betriebssystem für mobile Geräte nimmt immer mehr an Bedeutung zu. Laut Statistik wird das OS-Android auf über 70 % der neu produzierten mobilen Geräte verwendet.

Smartphone Betriebssysteme:
Marktanteile in Deutschland [%], Stand: Januar 2015

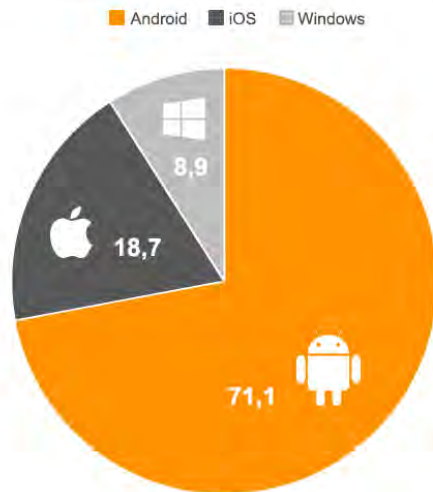


Abbildung 1: Marktanteil des Betriebssystems Android im Juli 2015

Das Konzept des Betriebssystems Android wurde im Jahr 2003 von dem gleichnamigen Unternehmen „Android“ entwickelt und im Jahr 2005 von dem Internetgiganten Google gekauft. Um die Weiterentwicklung von Android zu unterstützen, gründete Google im November 2007 zusammen mit mehr als 30 weiteren Unternehmen die Open Handset Alliance. Die erste Android-Version wurde im Oktober 2008 veröffentlicht und entwickelte sich schnell zu einem mächtigen Konkurrenten für iOS- und Windows Mobile-Betriebssysteme. Der Erfolg und die Popularität von Android beruhen auf der Leistungsfähigkeit des Software Development Kits von Android und der gepflegten quelloffenen Entwicklung. [1]



Abbildung 2: Smartphone-Apps auf dem Boardcomputer eines Fahrzeugs

Mit „Android Auto“ bietet Google auch für die Autoindustrie ein sicheres, einfaches und intuitives System, mit dem eine Fahrt mit zahlreichen attraktiven Features unterstützt wird. Darüber hinaus ist es mit „Android Auto“ möglich, eigene Apps, die auf dem Android betriebenen Smartphone installiert werden, auch auf dem Boardcomputer direkt im Fahrzeug über die Sprachsteuerung zu steuern. [2]

Das Teilsystem „Fahrzeugrechner“ des Projekts „Sauberes Fahren“ wird auf einem Tablet mit dem Betriebssystem Android entwickelt und unterstützt nicht alle Funktionalitäten eines modernen Boardcomputers. Die Arbeit fokussiert sich auf die Unterstützung eines Fahrers bei einer umweltfreundlichen und effizienten Fahrweise. Im nächsten Abschnitt wird ein grober Überblick über die Funktionalitäten des Teilsystems „Fahrzeugrechner“ gegeben.

*Diese Arbeit wurde durchgeführt bei der Firma IT-Designers, Esslingen

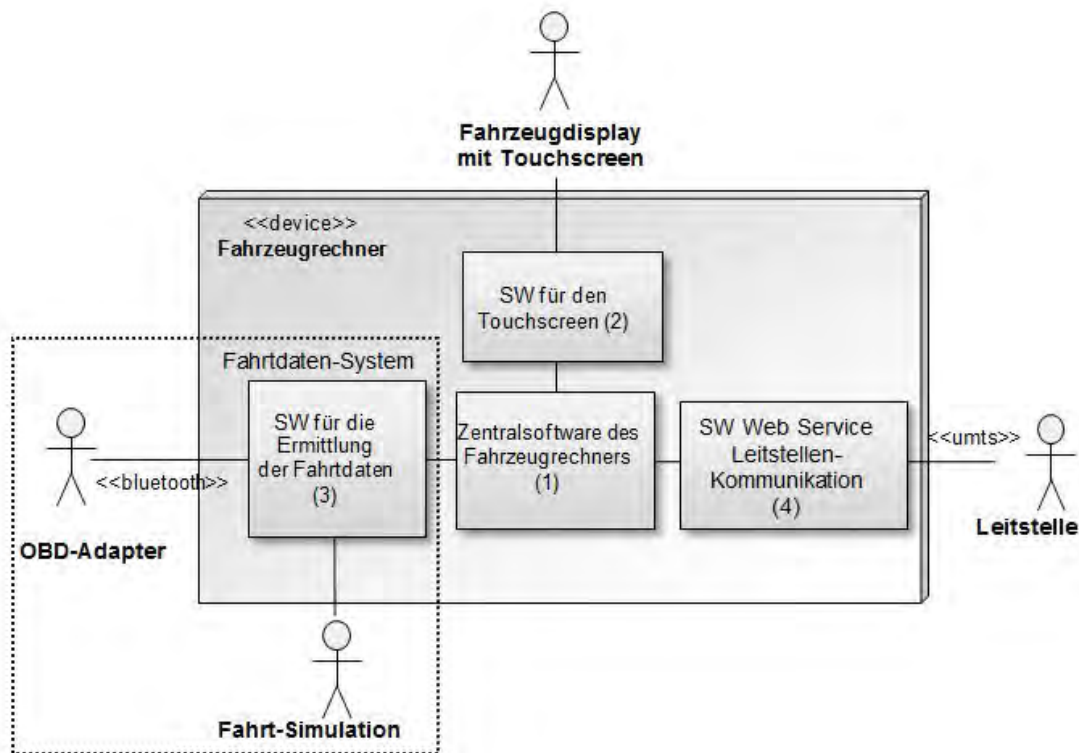


Abbildung 3: Softwarearchitektur des Teilsystems „Fahrzeugrechner“

Der Fahrzeugrechner ermittelt anhand von Daten, die während einer Fahrt gesammelt werden, den aktuellen Schadstoffausstoß, den mittleren Schadstoffausstoß, die aktuelle Fahrgeschwindigkeit und den aktuellen Kraftstoffverbrauch von Fahrzeugen und zeigt diese Werte während der Fahrt auf dem Display des Fahrzeugrechners an. So werden dem Fahrer alle wichtigen Anhaltspunkte zu einer effizienten und umweltfreundlichen Fahrweise gegeben. Um besonders umweltschädliche Fahrzeuge zu ermitteln, werden zu Beginn der nächsten Fahrt die mittleren Größen des Schadstoffausstoßes, die Fahrgeschwindigkeit und die VIN-Nummer an die Leitstelle übertra-

gen. Anhand von diesen Daten werden Statistikdaten aller Fahrzeuge erfasst, ausgewertet und dargestellt.

Der Kraftstoffverbrauch und die Fahrgeschwindigkeit werden mit Bordmitteln ermittelt. Dazu wird das Fahrzeugdiagnosesystem OBD genutzt. Alternativ zu Bordmitteln wird dem Fahrzeugrechner ein Simulationssystem zur Verfügung stehen, aus welchem sich die benötigten Fahrdaten in gleichem Format auslesen lassen. Dadurch kann das Gesamtsystem auch ohne ein echtes Kraftfahrzeug mit einigermaßen repräsentativen Daten getestet werden.

[1] <http://www.connect.de/ratgeber/android-geschichte-des-erfolgs-1491130.html>

[2] <http://www.golem.de/news/app-google-startet-android-auto-1503-113074.html>

Bildquellen:

- Abbildung 1: http://www.sanamedia.de/mobile_health.html
- Abbildung 2: lexusenthusiast.com/images/weblog/15-02-23-lexus-no-carplay-android-auto.jpg
- Abbildung 3: Eigene Darstellung

Evaluierung verschiedener Datenbanken im Embedded bereich

Thorsten Mollenkopf*, Harald Melcher, Jürgen Nonnast

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016



Abbildung 1: Embedded Datenbanken: Zeitmessung

Datenbanken sind heutzutage nicht mehr wegzudenken, da immer größere Datenmengen verarbeitet und gespeichert werden müssen. Datenbanken helfen zum Beispiel enorm Daten eines Anwendungsprogramms geordnet abzulegen und Verbindungen zu anderen Dateninformationen herzustellen. Ein Datenbanksystem besteht hierbei aus der Datenbank (DB) an sich und dem Datenbankmanagementsystem (DBMS). Das Managementsystem ist eine Software für den Benutzer, der so eine Datenbank erstellen und benutzen kann. Wichtigste Eigenschaft einer Datenbank ist das sogenannte **ACID-Paradigma**.

A: Als Atomicity wird eine Transaktion angesehen, da sie entweder komplett oder gar nicht ausgeführt wird.

C: Consistency bezeichnet den Zustand der Datenbank nach einer Transaktion. Sollte dies nicht der Fall sein, wird die letzte Transaktion rückgängig gemacht (roll-back).

I: Isolation ist die Bedingung, dass bei mehreren Transaktionen, die parallel ausgeführt werden, sie sich nicht gegenseitig beeinflussen dürfen.

D: Bei der Durability ist die Datenbank bei Abschluss einer Transaktion festgeschrieben, d.h. die Transaktion kann nicht rückgängig gemacht werden [1].

Gerade im Embedded Bereich werden nochmals besondere Anforderungen an eine Datenbank gestellt. Als aller Erstes sind auf einer Hardware mit eingebettetem Betriebssystem die verfügbaren Ressourcen sehr beschränkt, d.h. für eine Applikation stehen nur wenig RAM und wenig Festpeicher zur Verfügung. Dies gilt natürlich auch für eine Datenbank, die man verwenden möchte. Hier darf nicht nur die Datenbank möglichst klein sein, auch das DBMS sollte möglichst klein sein. Um dies zu erreichen wird bei dem Managementsystem meist Abstriche in Form der möglichen Optionen gemacht. Es werden nur Funktionen umgesetzt, die für die spätere Verwendung notwendig sind. Funktionen können zum Beispiel das Anlegen, Auslesen oder Löschen von Datenpaketen sein. Zusätzliche Funktionen wie zum Beispiel das Löschen von Doppelten Einträgen, das automatische Addieren bestimmter Datenpakete, das Verändern von Daten oder ähnliches sind meist optional. Eine weitere wichtige Bedingung ist die automatische Wiederherstellungsfunktion der Datenbank, denn gerade im Embedded Bereich muss mit Stromunterbrechungen oder Systemcrashes gerechnet werden. Hierbei können die Daten in der Datenbank oft sehr wichtig für die Analyse des Absturzes sein, wie zum Beispiel Fehler-Einträge. In der Praxis werden oft auch nicht rationale Datenbanken bevorzugt, da es sich meist nur um Key-Value-Daten handelt, die abgespeichert werden sollen. Bei diesem Projekt in Form einer Bachelorarbeit werden Datenbanken näher untersucht. Die ausgesuch-

*Diese Arbeit wurde durchgeführt bei der Firma Harman Becker Automotive GmbH, Stuttgart-Vaihingen

ten Datenbanken werben mit ihrer Größe und sind meist gerade für den Embedded Bereich konzipiert worden. Wichtig für die Entscheidung ist, wie die Datenbank zur Verfügung steht. Das zu untersuchende System soll unter einer öffentlichen Lizenz zur Verfügung stehen, d.h. der Sourcecode soll einsehbar und veränderbar sein, es soll möglichst kostenfrei sein und es sollen keine weiteren Bedingungen bei der Benutzung entstehen. Bei der Untersuchung der Datenbanken werden mehrere Eigenschaften untersucht. Hauptsächlich werden die Geschwindigkeit und Performance für eine bestimmte Aufgabe, aber auch die Benutzbarkeit, das Erstellen des Projektes und auch der Support und die Community hinter einem Produkt beleuchtet. Dies soll die Entscheidung für oder gegen eine bestimmte Datenbank erleichtern.

Untersuchte Datenbanken sind:

VedisDB: Eine C-Datenbank, die direkt auf Festplatte in eine Datei schreibt, was sehr wenig RAM benötigt. Sehr gut für ARM-Prozessoren geeignet. Verfügt über keine Netzwerkschnittstelle [2].

LevelDB: Eine C++-Datenbank, die hauptsächlich für Key-Value Daten eingesetzt wird. Es ist keine SQL-Datenbank, d.h. es werden keine SQL-Kommandos verwendet. Besonderheit ist eine Automatische Key-Sortierung, die die Applikation aber kaum verlangsamt [3].

LMDB: Ist eine C-Bibliothek Datenbank, deren innerer Aufbau eine B+ - Baumstruktur aufweist. Besonderheit ist die parallele Schreib- und Leseeigenschaft, da sich beide Vorgänge nicht gegenseitig blocken [4].

SQLite: Bekannteste Datenbank in dieser Auswahl. Eine relationale Datenbank, die unter mehreren APIs in C und C++ zur Verfügung steht. Gesteuert wird die Datenbank über die gängigen SQL-Kommandos und bietet viele Extrafunktionen, die das Benutzen der Datenbank vereinfacht [5].

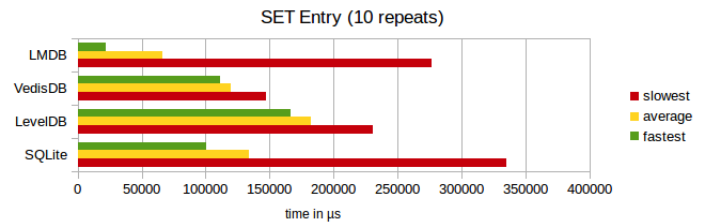


Abbildung 2: SET-Entry Funktion im Vergleich

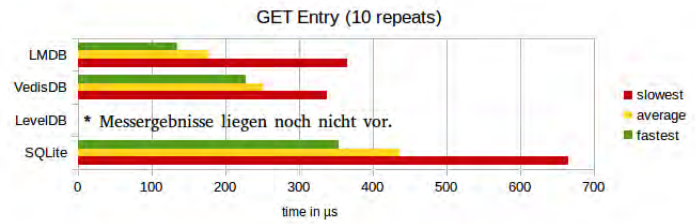


Abbildung 3: GET-Entry Funktion im Vergleich

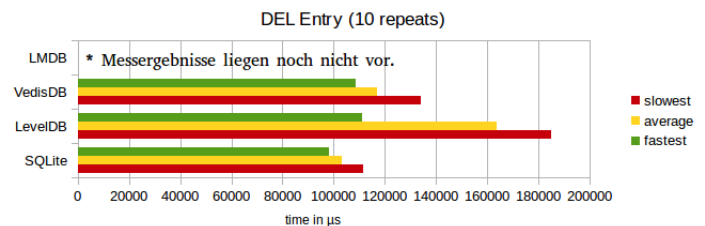


Abbildung 4: DEL-Entry Funktion im Vergleich

-
- [1] RAMEZ A. Elmasri und SHAMKANT B. Navathe, Grundlagen von Datenbanksystemen, Pearson Studium, 2009
 [2] <https://de.wikipedia.org/wiki/SQLite>
 [3] <http://vedis.symisc.net/>
 [4] https://en.wikipedia.org/wiki/Lightning_Memory-Mapped_Database
 [5] <https://en.wikipedia.org/wiki/LevelDB>

Bildquellen:

- Abbildung 1: <http://www.digi.com/products/embedded-systems/single-board-computers/connectcore-6-sbc/product-images/connectcore-6-sbc>
- Abbildung 2: made by Thorsten Mollenkopf
- Abbildung 3: made by Thorsten Mollenkopf
- Abbildung 4: made by Thorsten Mollenkopf

Konzipierung und Realisierung eines Übertragungsmechanismus zum Austausch von Steuerbefehlen und Daten zwischen Geräten für den kathodischen Korrosionsschutz und einer zentralen Leitebene basierend auf UMTS und unter Verwendung der Übertragungsprotokolle SNMP und HTTP

Nikolas Porzer*, Karlheinz Höfer

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Eine solide Grundversorgung mit Wasser, Öl und Gas, befördert über lange Distanzen, ist in der heutigen Zeit selbstverständlich. Immer längere Pipelines werden benötigt und müssen gegen die Witterung gesichert werden, um, neben der Gefährdung der Sicherheit und Umwelt, auch den jährlichen Schaden von rund 3 % des BIP entgegen wirken zu können [1]. Für den Schutz der Rohre wird auf kathodischen Korrosionsschutz (KKS) mit Fremdstrom gesetzt. Hierbei wird eine Schutzspannung auf die Leitung eingeprägt. Durch den elektrischen Widerstand der Pipelines und der Umgebung ist eine erneute Einprägung in regelmäßigen Abständen entlang des Rohres erforderlich. Da jedoch nicht an jedem Standort dieselben Umwelteinflüsse gleichermaßen auf das Rohr einwirken, muss für jeden Ort eine spe-

zifische Spannung gewählt werden. Um diese Arbeit den zuständigen Technikern zu erleichtern, werden seit vielen Jahren Fernüberwachungstechniken zur Steuerung und Messwerterfassung eingesetzt. Durch den Einsatz sogenannter SmartKKS-Systeme erweitert die RBS wave GmbH diese Praktik drastisch. Wie aus der Bezeichnung hervorgeht, bieten diese Geräte mehr als nur Fernüberwachung. Eine ständige Überwachung und Auswertung der KKS-Messdaten der angeschlossenen Rohrleitung und eine automatisierte Erkennung von Fremdeingriffen inklusive Absenden einer Alarmmeldung bietet dem Techniker einen deutlichen Mehrwert bei der Durchführung seiner Tätigkeit. Das Funktionsprinzip des SmartKKS-Systems ist in Abbildung 1 dargestellt.

Funktionsprinzip des Smart KKS-Systems



Abbildung 1: Funktionsprinzip des SmartKKS-System

Um diesen Mehrwert kontinuierlich leisten zu können, müssen die Geräte regelmäßig an den Bedarf von neuen Funktionalitäten angepasst werden. Hierfür sind Updates und neue Software für die Geräte unabdingbar. Eine Möglichkeit zur Installation neuer Software wäre der Ausbau der Geräte in Verbindung mit einer lokalen Ausführung der Erweiterungen. Stellt diese Möglichkeit bei geringer Anzahl von Geräten eine angemessene Option dar, so bietet sie für eine große Anzahl, aufgrund ei-

nes enormen Wartungsaufwands, keine geeignete Alternative. Benötigt wird daher ein Mechanismus für ein Update, welcher ebenso per Remote-Zugriff ausgeführt werden kann.

Wie zu Beginn angemerkt, stehen die SmartKKS-Systeme entlang der zu schützenden Pipelines, welche nicht selten weit entfernt von anderen Infrastrukturen verlaufen. Es kann somit keine Anbindung per Kabel an das Internet, wie es in Firmen oder Zuhause üblich ist, gewährleistet werden. Um dennoch ei-

*Diese Arbeit wurde durchgeführt bei der Firma RBS wave GmbH, Stuttgart und in Kooperation mit der Steinbeis Embedded Systems Technologies GmbH

nen Zugriff auf die Geräte zu erlangen, wird das mobile Datennetz genutzt. Da die Verbindungsqualität in Abhängigkeit von Ort, Anzahl verfügbarer Sendemasten und Teilnehmern innerhalb derselben Zelle sehr stark schwankt, stellen die Übertragung und somit auch der Remote-Zugriff über das mobile Datennetz eine große Herausforderung dar. Es ist ein Protokoll erforderlich, welches mit den Nachteilen der mobilen Datennutzung weitestgehend umgehen und somit für eine zuverlässige Kommunikation sorgen kann.

Im Rahmen der Bachelorarbeit wurden zunächst typische Daten sowie deren Größe für die Übertragung zu den einzelnen Systemen festgelegt und anhand dieser unterschiedliche Übertragungsprotokolle für den Einsatz untersucht. Für die Steuerung des Übertragungsmechanismus werden Befehle mittels *Simple Network Management Protocol (SNMP)* an das SmartKKS-System gesendet und von diesem ausgewertet. Je nach abgesetztem Befehl wird ein Download von einem zentralen Server, die Installation eines heruntergeladenen Update-Pakets oder die Erstellung und Bereitstellung eines Logfiles über den Webserver des Gerä-

tes angestoßen. Für die Datenübertragung sind das *Secure Copy Protocol (SCP)*, das *Secure File Transfer Protocol (SFTP)* und das *Hypertext Transfer Protocol (HTTP)* von Bedeutung und wurden speziell in Hinsicht auf Übertragungsgeschwindigkeit, anfallenden Overhead und die Möglichkeit der Wiederaufnahme unterbrochener Übertragungen geprüft. Um den Overhead und die Übertragungsgeschwindigkeit für alle drei ausgewählten Protokolle gleichermaßen zu testen, sind vier Testdateien mit den zuvor festgelegten typischen Größen mit Zufallswerten erzeugt worden. Sie wurden wiederholt mit demselben Protokoll übertragen, um dadurch Seiteneffekte, beispielsweise eine starke Belastung des Netzwerkes durch andere Teilnehmer, zu minimieren. Parallel dazu fand eine Aufzeichnung mit dem Diagnosewerkzeug Wireshark statt, welche anschließend ausgewertet und mithilfe von Excel grafisch aufbereitet wurde. Dadurch konnte ein direkter Vergleich ermöglicht werden. Abbildung 2 zeigt den Vergleich des relativen Overheads aller drei Protokolle in Abhängigkeit von den übertragenen Daten.

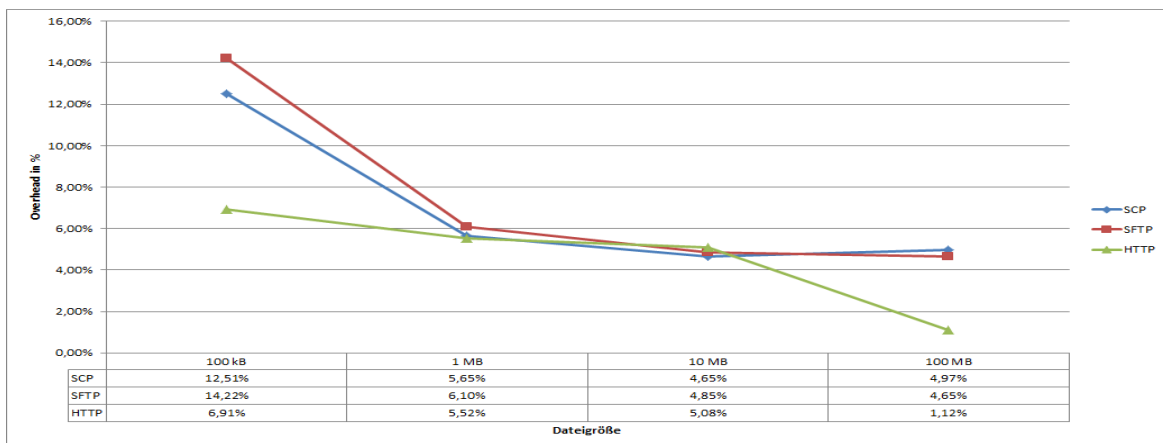


Abbildung 2: Übersicht über den prozentualen Overhead der einzelnen Protokolle in Abhängigkeit der Dateigröße

Nach Festlegung des *Hypertext Transfer Protocols* als Übertragungsprotokoll und Konzeptionierung der einzelnen Abläufe mithilfe von Sequenzdiagrammen, konnten die dafür notwendigen internen Abläufe und Strukturen für die Ansteuerung der einzelnen Funktionen innerhalb des SmartKKS- Systems an-

gelegt werden. Die Übertragung wird durch die Verwendung der *libcurl*-Bibliothek, welche das ausgewählte Protokoll unterstützt und eine saubere Umsetzung garantiert, realisiert [2]. Abschließend stehen die finale Umsetzung und die Verifikation der Implementierung aus.

[1] Donner, S: Rost vernichtet drei Prozent des Bruttoinlandsprodukts. Handelsblatt (2009) [online] <http://www.handelsblatt.com>

[2] <http://curl.haxx.se/libcurl/>

Bildquellen:

- Abbildung 1: RBS wave GmbH
- Abbildung 2: Eigene Darstellung

Prototypische Realisierung einer closed-loop Testumgebung für eine Dosing-Control-Unit (SCR System) auf Basis *µLabCar*

Martin Pröbstle*, Reiner Marchthaler, Werner Zimmermann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In der Automobilindustrie stellen Fahrzeughersteller (*OEM*) den Zulieferern Versuchsträger zur Verfügung. Diese werden während des Applikationsprozesses vielfältigen Tests unterzogen, um eine Freigabe für die Serienfertigung zu erhalten. Zeit- und Kostendruck nehmen dabei ständig zu. Daraus folgt, dass inzwischen immer weniger Versuchsträger aufgebaut werden, was für den *OEM* eine erhebliche Kostenersparnis darstellt. Dies wiederum bewirkt eine Verkürzung der Zeiträume, in welchen der *OEM* seine Versuchsträger zur Verfügung stellt. Bedingt durch die genannten Umstände erhalten Zulieferer weniger Testmöglichkeiten, um Steuergerätfunktionen zu applizieren und zu validieren. Dies erfordert immer effizientere Arbeitsprozesse und die Entwicklung von Alternativen, um den Anforderungen an Qualität gerecht werden zu können, ohne dabei auf Versuchsträger angewiesen zu sein. Eine Möglichkeit stellen sogenannte *FullLabCars* dar. *FullLabCars* ermöglichen das Testen einzelner Komponenten und Funktionen eines Fahrzeugs ohne Versuchsträger. Ihr Einsatz birgt jedoch gleichzeitig Herausforderungen, wie hohe Anschaffungskosten, eine komplexe Inbetriebnahme sowie Wartung. Dadurch bleiben Testsysteme mit *FullLabCars* für einen fest vorgeschriebenen Zeitraum in Betrieb. Die Länge des Zeitzyklus wird entsprechend des Umfangs der Testanforderung bestimmt. Dies schränkt die Flexibilität für die Veränderung eines Testaufbaus signifikant ein. Das macht das *FullLabCar*-System gerade für kleinere Testfälle unattraktiv [1].

Die *Bosch Engineering Group (BEG)* hat dies als Anlass zur Entwicklung eines schlanken Testsystems genommen.



Abbildung 1: BEG *MicroLabCar*

Ihr entwickeltes *MicroLabCar* soll eine hohe

Flexibilität und einen geringen Aufwand bei Inbetriebnahme und Wartung bieten. Es verfügt dabei über die wichtigsten Schnittstellen, die zur Kommunikation und zur Applikation eines Steuergerätes notwendig sind. Dazu gehören analoge und digitale Ein/Ausgänge, *PWM*-Signale sowie Busschnittstellen, wie *SENT*, *CAN*, *LIN*, *Flexray* und *PSI5*. Das *MicroLabCar* wird durch eine Anwendung auf einem *Windows-PC* gesteuert. Über diese können dann die einzelnen Pins auf der Hardware angesprochen werden. Das *MicroLabCar* ist für *open-loop* Tests ausgelegt, was bedeutet, dass nur das Eingangssignal ins System betrachtet wird und keine Regeldifferenz aus Eingangs- und Ausgangssignal gebildet wird [2].

Im ersten Teil der Arbeit wurde das *MicroLabCar* in Verbindung mit einem Steuergerät und einer Spannungsversorgung in Betrieb genommen, um einen Verbindungsaufbau über die *.NetApi* zwischen *MATLAB* und dem *MLC* zu testen. Durch diese Tests wurden die Möglichkeiten und die Performance der *.NetApi* überprüft, da sie einen Hauptteil dieser Arbeit darstellten.

Die Hauptaufgabe der Arbeit bestand darin einen geschlossenen Regelkreis (*closed-loop*) zu realisieren. Dieser Regelkreis wurde im Bereich der Abgasnachbehandlung gewählt. Hier bietet die *Robert Bosch GmbH* als eine Möglichkeit das *Selective Catalytic Reduction (SCR)*-Verfahren an. Im *SCR*-System befinden sich die Komponenten Tank, Pumpenmodul, Dosiermodul und *Dosing Control Unit (DCU)*. Im Tank wird die Harnstofflösung *AdBlue* gelagert. Sobald das Fahrzeug gestartet wird erfolgt ein Systemhochlauf. Dabei beginnt das Pumpenmodul den vorgegebenen Solldruck aufzubauen. Nach dem Hochlauf ist das System betriebsbereit. Es kann nun durch das Öffnen des Dosiermoduls *AdBlue* in den Abgasstrang eindosiert werden. Die Dosiermenge ist hierbei abhängig von Temperatur und der zu diesem Zeitpunkt anfallenden *NOx*-Emissionsmenge im Abgasstrang [3]. Dieses System eignet sich sehr gut für den Versuch einen geschlossenen Regelkreis über das *MicroLabCar* zu realisieren, da es nur einen Regelkreis gibt und somit weniger

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Stuttgart-Feuerbach

Abhängigkeiten existieren. Der Regelkreis ist für die Druckregelung zwischen Pumpenmodul und Dosiermodul zuständig. Dabei galt es sowohl den Druck während des Aufbaus, als auch während des Eindosierungsvorgangs zu regeln. Als Eingang dienten die Sollgröße, der Soll-Systemdruck und die Regelgröße Ist-Systemdruck. Die sich dabei ergebende Regelabweichung wurde von der *DCU*, in welcher sich der Regler befand, erfasst und verarbeitet. Die *DCU* stellt einen *Duty-cycle* ein,

der die Drehzahl des Motors im Pumpenmodul beeinflusst und somit den Druck am Dosiermodul reguliert. Da es nicht möglich war dieses System am Schreibtisch physikalisch zur Verfügung zu stellen, wurde ein entsprechendes Modell, welches das physikalische Verhalten des Systems abbildete, in *MATLAB* implementiert. Der in *MATLAB* berechnete Ist-Druck wurde an den entsprechenden Pin des *MicroLabCar* ausgegeben. Folgende Hardwarekomponenten wurden eingesetzt:

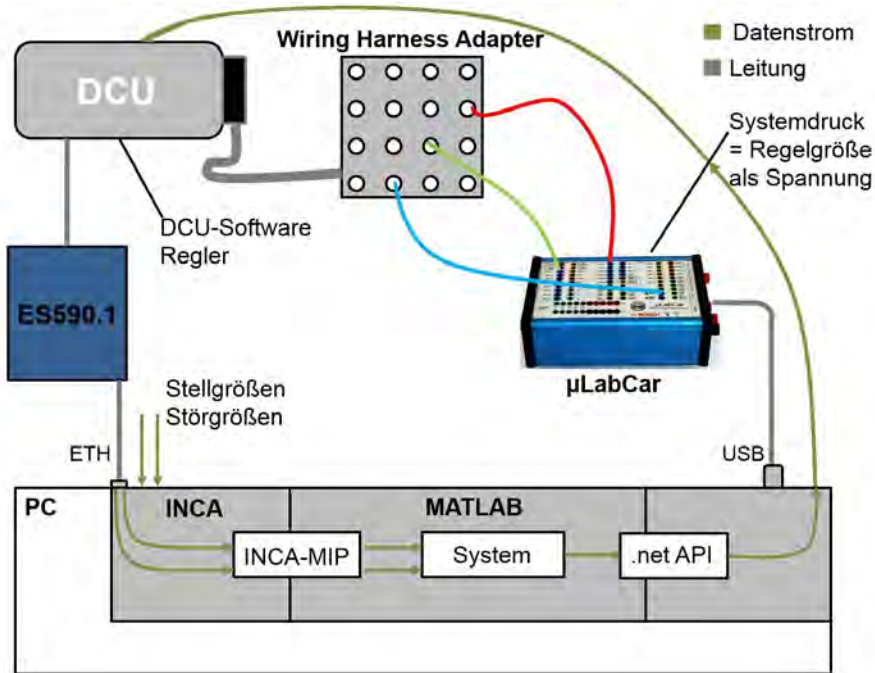


Abbildung 2: Schematischer Aufbau

- *Laptop*: Plattform für *MATLAB* Modell und für Mess-/Applikationssoftware *INCA*.
- *ETK*: Kommunikationsschnittstelle zwischen *Laptop*(*INCA*) und *DCU*.
- *MicroLabCar*: Stellen der Ausgangswerte über die verfügbaren Pins.
- *Wiring Harness Adapter*: Ist die Verbindung des Kabelbaums der *DCU* zu den Pins des *MicroLabCar*.
- *DCU*: Steuergerät mit integrierten Regler zur Regelung des in *MATLAB* implementierten Modells. Verbunden mit *ETK* und *Wiring Harness Adapter*.

[1] Bosch Intern(2015)
 [2] Beschreibung BEG MicroLabCar (2015)
 [3] Kraftfahrzeugtechnisches Taschenbuch 28. Auflage(2014)

Bildquellen:

- Abbildung 1: Bosch Calibration Wiki MLC
- Abbildung 2: Bosch Intern

Integration von Smart Home Geräten in das homee System

Tankred Raitz*, Werner Zimmermann, Karlheinz Höfer

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Vernetzung und Automatisierung sind zwei Begriffe, die seit einigen Jahren immer mehr an Bedeutung gewinnen. Im Bereich der Wirtschaft ist das Thema Digitalisierung unter dem Schlagwort Industrie 4.0 zusammengefasst und beschreibt eine neue Stufe der industriellen Fertigung. Einzelne Maschinen können nun miteinander kommunizieren um den ganzen Ablauf der Fertigung zu optimieren.

Die Automatisierung und Vernetzung in Wohnungen und Häusern wird unter dem Begriff „Smart Home“ zusammengefasst. Es geht hierbei um eine vollvernetzte intelligente Gebäude-/Haustechnik. Außerdem trägt „Smart Home“ dazu bei, den Klimawandel, Ressourcenknappheit und das Älterwerden der Gesellschaft zu bewältigen [1]. Das Ziel ist es, einerseits die Lebensqualität zu erhöhen, das Leben sicherer zu gestalten und andererseits Energie einzusparen. Erreicht wird dieses Ziel durch die sich aus der Vernetzung aller Geräte ergebenden Möglichkeiten. Nicht nur Lampen, Rollläden und Schalter sind verknüpft, sondern auch die Heizungssteuerung und Geräte wie Waschmaschinen oder der Herd. Außerdem können viele verschiedene Sensoren wie Temperatur-, Bewegung-, CO₂-, Wasser-, sowie Tür- und Fenstermelder miteingebunden werden. Aus den Sensordaten können durch die Vernetzung viele Abläufe automatisiert werden. Ein einfaches Beispiel ist das automatische Ausschalten der Heizung, sobald ein Fenster geöffnet wird. Durch eine intelligente Heizungssteuerung kann viel Energie gespart werden.

Auf dem Smart Home Markt ringen viele Hersteller um Kunden. Sie bieten viele verschiedene Produkte an und setzen auf unterschiedliche Funkprotokolle. Wichtige Protokolle sind Z-Wave, EnOcean, ZigBee, WLAN und Bluetooth. Einige Hersteller nutzen diese Standards, andere verwenden jedoch eigene geschlossene Systeme. Der Stromverbrauch ist für viele Geräte entscheidend, was die Wahl des Funkprotokolls erschwert.

Aus dieser Vielfalt ergibt sich für den Kunden das Problem, dass sich die Geräte eines Herstellers nicht mit denen eines anderen Anbieters vernetzen lassen. Dieses Problem erkannte unter anderem die Firma Codeatelier in

Stuttgart und entwickelte das Smart Home System „homee“.



Abbildung 1: homee Würfel

homee besteht aus einzelnen Würfeln, welche kombiniert werden können. Jeder Würfel unterstützt eine Funktechnologie. Dadurch ist homee mit vielen Herstellern und Produkten kompatibel. Außerdem ist das System erweiterbar, sollten sich in Zukunft neue Protokolle durchsetzen.

Diese Bachelorarbeit beschäftigt sich mit der Integration weiterer Geräte in das bestehende homee System. Bei den Geräten handelt es sich um den lernfähigen Thermostat und den Rauch- und CO-Melder der Firma Nest.

Nest Labs wurde 2010 vom iPod-Erfinder Tony Fadell und Matt Rogers gegründet. Vier Jahre später kaufte Google das Unternehmen für 3,2 Mrd US-Dollar. Mittlerweile wird die dritte Generation des Raumthermostates auch außerhalb der USA verkauft. Das Konzept von Nest sieht eine dauerhafte Internetverbindung der eigenen Produkte (Thermostat, Rauchmelder, Kamera) vor. Dadurch kann man von überall mittels der Nest-App sehen, ob Zuhause alles in Ordnung ist.

Für Entwickler bietet Nest eine Cloud-API an. Über diese Schnittstelle können die meisten Funktionen der Geräte angesprochen werden. Für die verschiedenen Geräte gibt es bestimmte Berechtigungen, denen der Nutzer einmalig zustimmen muss. Diese Cloud-API bietet die Grundlage der Integration in das homee-System. Die Aufgabe ist es, alle verfügbaren Funktionen der API zu nutzen. Der erste Schritt

*Diese Arbeit wurde durchgeführt bei der Firma Codeatelier GmbH, Ostfildern

ist die Autorisation, welche über OAuth2 abgewickelt wird. Hier findet die Verknüpfung des Nutzers mit der Nest-Applikation statt. Der zweite Teil besteht darin, die vorhandenen Geräte (Thermostate und Rauchmelder), sowie deren Werte (Temperatur, Rauchalarm, usw.) anzuzeigen. Für die Anfragen an die Nest-Cloud-API wird die REST (Representational State Transfer) Architektur verwendet. Die Kommunikation findet dabei über das HTTPs-Protokoll statt. Hierfür soll die C++-Bibliothek von cURL (Curl URL Request Library) verwendet werden, da diese bereits im homee-System eingesetzt wird. Den Geräten wird ein Profil zugeordnet sowie für jeden Anzeigewert ein entsprechendes Attribut. Durch diese Vorgehensweise wird die Unterstützung von vielen verschiedenen Geräten erleichtert, da die Profile und Attribute wiederverwendet und neu kombiniert werden können.

Die größte Aufgabe besteht darin, die Werte bei einer Änderung zu aktualisieren. Eine Möglichkeit ist das zyklische Abfragen („polling“) der Werte. Allerdings begrenzt Nest die Anfragen an die Cloud-API, um die Server nicht zu überlasten. Dadurch entsteht eine Verzögerung und die angezeigten Werte sind nicht immer aktuell. Als Alternative bietet Nest eine REST-Streaming genannte Funktion. Diese basiert auf Events, welche vom Server gesendet werden. Diese „Server-sent events“ sind ein Bestandteil der HTML5 Spezifikation. Ein-

ordnen lässt sich die Funktion zwischen einer normalen HTTP-Anfrage, welche mit der Serverantwort abgeschlossen ist, und Websockets, bei denen die Verbindung bestehen bleibt und Daten empfangen und gesendet werden können. Mit Server-sent events kann nur der Server Daten senden. Diese sind in bestimmten Events definiert, wie „keep-alive“ oder „put“, falls sich die Werte geändert haben. Die Änderungen werden live mitgeteilt, was der große Vorteil gegenüber Polling ist.



Abbildung 2: homee mit Nest Thermostat

Die Bachelorarbeit erweitert die unterstützten Geräte von homee um die Produkte von Nest. Dadurch können diese besser mit anderen Geräten im smarten Zuhause kombiniert und einfach über die homee-App bedient werden.

[1] <http://www.zertifizierungsprogramm-smarhome.de/smarhome/programm/seiten/hintergrundundmotivation.aspx> Abruf: 25.11.2015

[2] <https://developer.nest.com/documentation> Abruf: 25.11.2015

Bildquellen:

- Abbildung 1: Codeatelier GmbH
- Abbildung 2: Tankred Raitz

Modellbasierte Längs- und Querregelung für ein autonom fahrendes Modellauto

Julian Rapp*, Reiner Marchthaler, Werner Zimmermann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Seit einigen Jahren rückt die Forschung im Bereich des autonomen Fahrens immer mehr in den Fokus großer Unternehmen. Nicht nur große Automobilhersteller wie Mercedes-Benz arbeiten an der Entwicklung autonomer Fahrzeuge, sondern auch Softwarehersteller wie Google. Aber auch in ganz anderen Bereichen, wie zum Beispiel im Katastrophenschutz, wird an autonomen Systemen aller Art geforscht. Doch alle autonomen Fahrzeuge, egal ob Auto oder Roboter, haben eins gemeinsam, um sich zu bewegen, brauchen sie einen Regler der dafür sorgt, dass das Fahrzeug auch die Position erreicht, die es erreichen soll.

Die Aufgabe der Arbeit besteht darin einen Längs- und Querregler für ein autonom fah-

rendes Modellauto zu entwickeln. Der Längsregler regelt die Geschwindigkeit, der Querregler das Lenkverhalten des Fahrzeugs. Hierdurch ist es möglich einen vorgegebenen Kurs auf der Basis einer gespeicherten Karte oder mittels Fahrspurerkennung durch eine Kamera autonom abzufahren.

Der Regler wird auf einem bereits existierenden Modellauto, bestehend aus zwei Raspberry PI und einem Arduino Due, implementiert. Das Arduinoboard liefert die notwendigen Sensorwerte Drehrate $\dot{\psi}$ und die Raddrehzahlen. Zwei Kalman-Filter schätzen aus diesen Größen die für die Regelung notwendigen Größen, Fahrzeuggeschwindigkeit und Gierwinkel ψ .

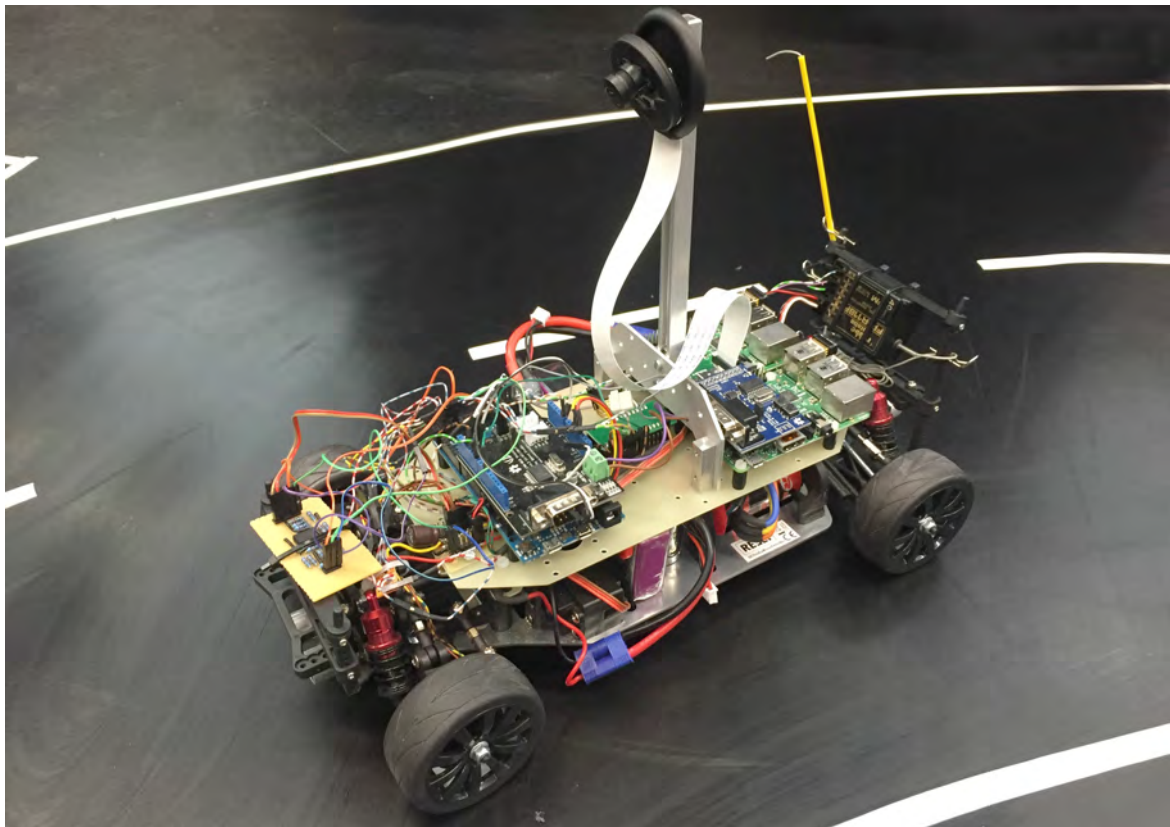


Abbildung 1: autonom fahrendes Modellauto

*Diese Arbeit wurde durchgeführt an der Fakultät Informationstechnik

Für die Querregelung bestimmt das Modellauto seine Position im Weltkoordinatensystem. Die Position bestimmt das Modellauto mit Hilfe des zurückgelegten Wegs und dem Gierwinkel ψ . Um nun einen Querversatz zu er-

mitteln vergleicht das Modellauto seine aktuelle Position mit der Position die es erreichen soll und bestimmt daraus den Querversatz. Entsprechend dem Querversatz wird dann mit Hilfe eines PI-Reglers die Lenkung angesteuert.

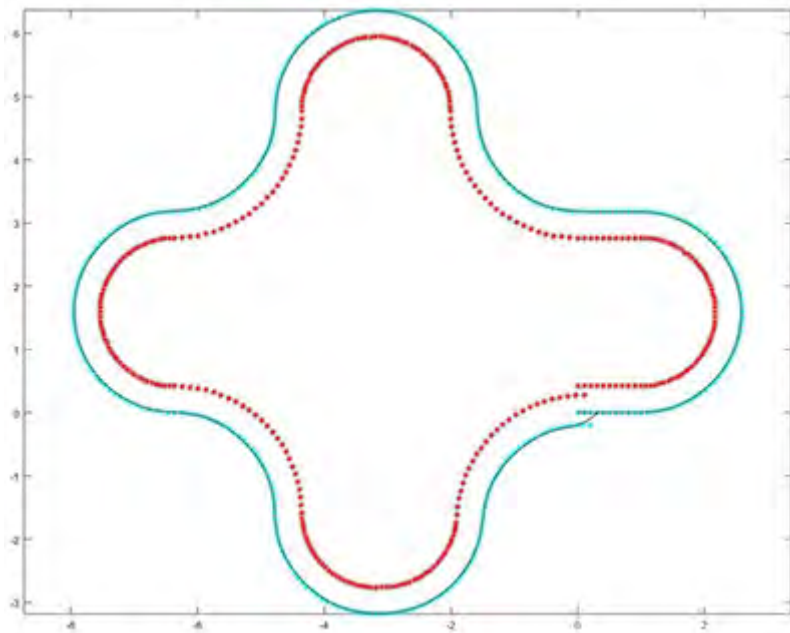


Abbildung 2: Beispielkurs

Eine Strecke besteht wie in Abbildung 2 dargestellt aus einer linken und einer rechten Spur. Die linke Spur ist in rot dargestellt, die rechte Spur ist in blau dargestellt. Die schwarze Linie zeigt die Strecke die das Modellauto mit Hilfe des Reglers gefahren ist.

Mit Hilfe einer Variablen kann die rote bzw. die grüne Spur gewählt werden, so dass der Regler das Fahrzeug der roten oder grünen Spur folgen lässt. Ebenso ist es möglich während der Fahrt die Spur zu wechseln. Die Abbildung 3 zeigt einen solchen Spurwechsel.

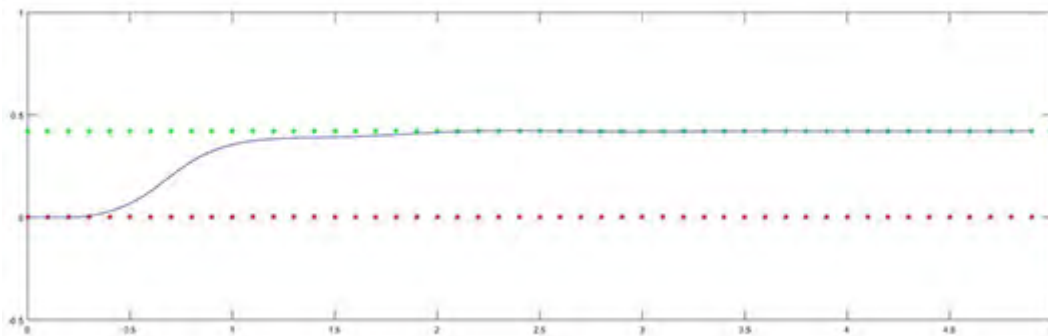


Abbildung 3: Spurwechsel

Bildquellen:

- Abbildung 1–3: Julian Rapp

Evaluation von OpenCL für FPGAs

Matthias Roth*, Peter Väterlein, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Field Programmable Gate Arrays (FPGAs) werden häufig als programmierbare Hardware beschrieben. Durch ihre Anpassungsfähigkeit an Probleme zeichnen sie sich in verschiedenen Anwendungen durch hohe Performanz bei guter Energieeffizienz im Vergleich zu CPU- und GPU-Lösungen aus. Bei der Geschwindigkeit und Energieeffizienz werden FPGAs jedoch meist von den nicht wiederprogrammierbaren anwendungsspezifischen Schaltungen (ASICs) übertroffen. Doch auch moderne FPGAs beinhalten immer mehr vorgefertigte Funktions-Blöcke, wie zum Beispiel arithmetische Recheneinheiten. Dies führt dazu, dass die Leistungsdifferenz zwischen FPGAs und ASICs kleiner wird, wodurch FPGA-Lösungen wirtschaftlich sinnvoller werden und ihr Einsatz in immer mehr Anwendungsgebieten attraktiv erscheint.

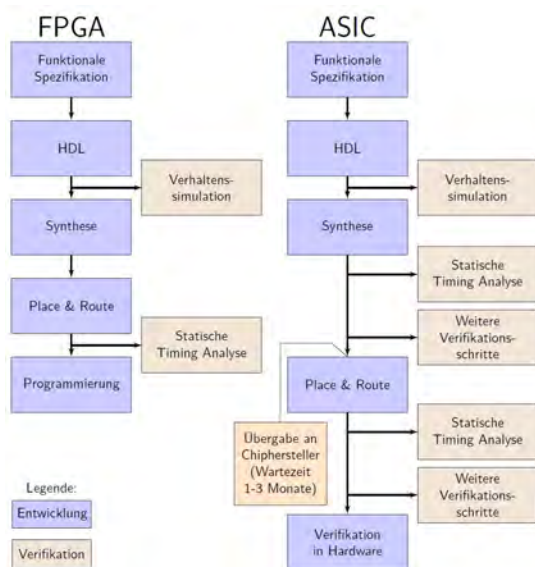


Abbildung 1: Entwicklung von FPGA- und ASIC-Designs im Vergleich

Die Anwendungsentwicklung für digitale Hardwarekomponenten wie FPGAs ist sehr komplex. Ein Algorithmus muss erst in einzelne Logikbausteine zerlegt werden, wozu detaillierte Kenntnisse der Hardware nötig sind und beachtet werden muss, dass die Ressourcen eines FPGAs begrenzt sind. Wie diese

Bausteine verwendet werden und welche Beziehungen zwischen diesen herrschen, wird in einer Hardwarebeschreibungssprache wie beispielsweise VHDL festgehalten. Daraufhin wird die beschriebene Logik in der Synthese an die Hardware des FPGAs angepasst und anschließend muss das Ergebnis der Synthese noch auf die vorhandene Hardware gelegt (Place) und verdrahtet (Route) werden. Nach diesem Schritt müssen bei zeitkritischen Anwendungen unter Umständen noch die Timings von einzelnen Verdrahtungen überprüft werden. ASICs sollten in der Entwicklung zusätzlich in mehreren Schritten verifiziert werden, da nachträgliche Änderungen mit hohen Mehrkosten verbunden sind, dadurch erhöht sich der Aufwand in der Entwicklung wie Abbildung 1 anschaulich zeigt. Der Einsatz von digitalen Hardwarekomponenten führt somit zu einer vergleichsweise hohen Entwicklungszeit, wodurch sie nur bei hohen Stückzahlen wirtschaftlich konkurrenzfähig zu CPU- und GPU-Lösungen sind.

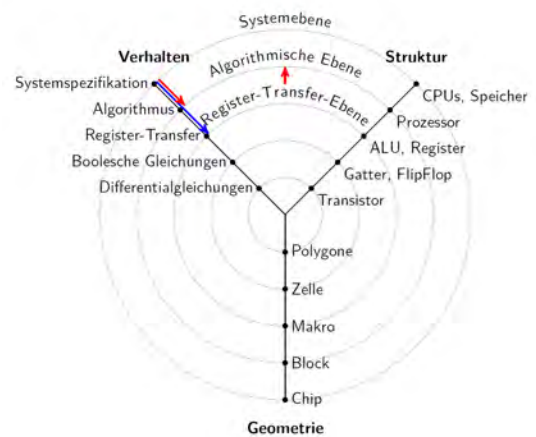


Abbildung 2: Y-Diagramm nach Gajski

In den 1980er Jahren wurde das Y-Diagramm (s. Abbildung 2) von Gajski und Kuhn entworfen, welches die verschiedenen Sichtweisen auf die Hardwareentwicklung beschreibt. Der Entwurf einer Anwendung erfordert bei der Verwendung von Hardwarebeschreibungssprachen ein Vorgehen, bei welchem Konzepte für alle Ebenen von der Sys-

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Renningen

temebene bis zur Register-Transfer-Ebene erstellt werden müssen (blauer Pfeil). Beispielsweise im Bereich der Bildverarbeitung tritt nun das Problem auf, dass digitale Hardwarekomponenten die Algorithmen stark beschleunigen, diese sich aber schnell ändern oder schnell ausgetauscht werden, sodass der Einsatz von ASICs oder FPGAs den Fortschritt einschränkt, da der Fokus auf die Hardwareentwicklung gerichtet wird und nicht auf das Problem. Es gibt bereits verschiedene Ansätze zur Abstraktion der Entwicklung nach Vorbild des Y-Diagramms (rote Pfeile). Meist werden Programmiersprachen verwendet, welche auf die sequentielle Verarbeitung von Befehlen ausgelegt sind. Die Parallelität der FPGAs lässt sich mit diesen Sprachen oft nur ineffizient nutzen, weshalb sich deren praxistaugliche Verwendung nur auf spezielle Anwendungsgebiete beschränkt.

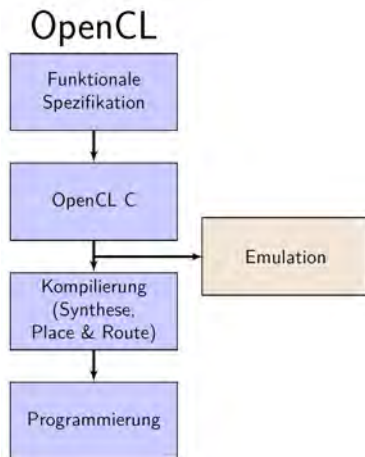


Abbildung 3: Entwicklung von FPGAs mit OpenCL

Die beiden führenden FPGA-Hersteller bieten nun mit OpenCL eine neue Programmiermethodik an, die ebenfalls auf ein Hochsprachenkonzept aufsetzt[1][2]. Es handelt sich dabei um ein Framework für hochgradig parallelisierbare Probleme. Das bedeutet für einen Entwickler, dass dieser das Verhalten nun als Algorithmus in einer C ähnlichen Sprache

beschreiben kann und kein Umdenken zu Logikbausteinen mehr nötig ist. Das entwickelte Programm kann am PC emuliert werden um die funktionale Richtigkeit sicherzustellen, bevor der Ziel-FPGA nach dem Kompilervorgang programmiert wird (s. Abbildung 3). FPGAs sollen dadurch bei dem in der Produktentwicklung wichtigen Maß *Time-to-Market* mit anderen Lösungen mithalten können.

Dieser Anspruch, eine hochperformante Lösung für ein komplexes Problem mit geringem Aufwand zu finden, wird in dieser Arbeit untersucht. Die Evaluation wird anhand eines Beispiels aus der Bildverarbeitung durchgeführt. Der gewählte Algorithmus beansprucht die Ressourcen von FPGAs in hohem Maße und eignet sich so hervorragend zur Untersuchung der OpenCL-Lösung. Durch den Vergleich mit einer bestehenden VHDL-Lösung kann der Entwicklungsprozess optimal verglichen werden. Dabei wird bewertet, wie sich die Entwicklungszeit verändert und welche Hardwarekenntnisse nötig sind, um eine ressourcenoptimierte Lösung zu erhalten. Anschließend wird die entstandene Anwendung nach verschiedenen Automotive-Kriterien wie Safety und Echtzeitfähigkeit bewertet.

Ziel der Arbeit ist die Beurteilung der Umsetzbarkeit von OpenCL auf FPGAs und des Reifegrades des Altera OpenCL SDKs. Zudem wird eine mögliche FPGA-Architektur speziell für den Einsatz von OpenCL untersucht. Es wird bestimmt welches Verhältnis zwischen der Logik, dem Speicher und den Rechen-einheiten bestehen sollte, wenn die FPGA-Anwendung mit OpenCL entwickelt wird. Dazu gilt es herauszufinden welche Strukturen der OpenCL-Compiler effizient umsetzen kann und welche Strukturen häufig genutzt werden. Schließlich soll die optimale FPGA-Architektur für die Bildverarbeitung mit OpenCL gefunden werden. Das Ergebnis wird außerdem Aufschluss darüber geben, ob OpenCL bereits im Automotive-Umfeld eingesetzt werden kann, beziehungsweise aufzeigen, welche Fortschritte noch erzielt werden müssen.

[1] Altera SDK for OpenCL: <https://www.altera.com/products/design-software/embedded-software-developers/opencl/overview.html>

[2] Xilinx SDAccel Development Environment: <http://www.xilinx.com/products/design-tools/software-zone/sdaccel.html>

Bildquellen:

- Abbildung 1: Eigene Darstellung nach <http://www.xilinx.com/fpga/asic.htm>
- Abbildung 2: Eigene Darstellung nach F. Kesel: Entwurf von digitalen Schaltungen und Systemen mit HDLs und FPGAs, De Gruyter Verlag, 2013
- Abbildung 3: Eigene Darstellung

Use Case 2.0 – Ivar Jacobson 2011

Lukas Ruckwied*, Manfred Dausmann, Joachim Goll

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Leichtgewichtig, anpassbar und vielseitig

„Use-Case 2.0, The Guide to Succeeding with Use Cases“ aus dem Jahre 2011 von Ivar Jacobson[1] beschreibt eine Technik, mit der die ehemals schwergewichtigen Use Cases (1.0) an eine agile und iterative Entwicklung angepasst werden können. Innerhalb eines Zyklus einer Iteration ist es kaum möglich, die Abläufe eines klassischen, schwergewichtigen Use Case samt Analyse, Implementierung und Test zu realisieren. Use Case 2.0 beschreibt einen Weg, um die bekannten Use Cases 1.0 in kleine iterative Scheiben zu teilen und damit für eine agile Entwicklung passend zu machen.

Der Kern des Konzepts von Use Case 2.0 ist, dass ein Use Case anhand von Stories vertikal in sogenannte Scheiben (engl. slices) aufgeteilt wird, um daraus mehrere kleine Planungseinheiten für die zyklische Auslieferung zu erzeugen. Diese Use Case Slices (dt. Anwendungsscheiben) können in agilen Entwicklungsmethoden den Platz der sogenannten User Stories einnehmen. Die aus Extreme Programming bekannten User Stories sind klein und damit bestens als Planungseinheit für die kurzen Zyklen einer iterativen Entwicklung geeignet. Use Case Slices, die ähnlich wie die User Stories aufgebaut sind und dedizierte Abläufe umfassen, enthalten darüber hinaus eine Referenz auf einen schwergewichtigen Use Case. Durch diese Referenz, wie in Abbildung 1 veranschaulicht, ist es möglich, eine Use Case Slice auf den zugehörigen Use Case zurückzuführen und damit in das klassische Use Case-Diagramm eines Systems einzuordnen. Damit erhält man eine Gesamtschau des Systems und ein gemeinsames Verständnis, auch „Shared Understanding“ genannt, wie das System aussehen soll. Beides ist für eine zielgerichtete Entwicklung eines Systems eine absolute Notwendigkeit. Bei der Verwendung von User Stories hingegen ist dieses Ziel nicht ohne erheblichen zusätzlichen Aufwand erreichbar.

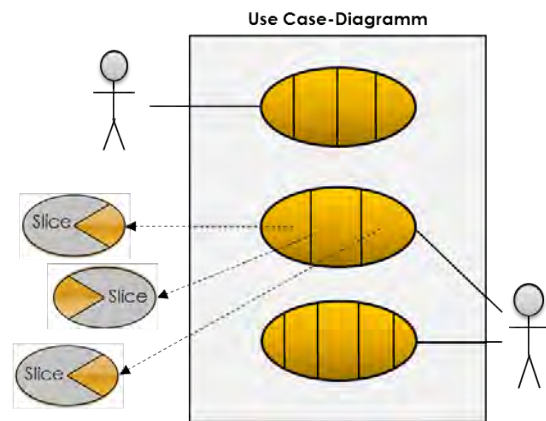


Abbildung 1: Gesamtsicht durch Use Case-Diagramm

Ein Use Case (1.0) sollte entsprechend seiner Abläufe aufgeteilt werden. Diese Aufteilung, auch Schneiden genannt, trennt einen Use Case entlang seiner Abläufe und wird somit „vertikal“ durchgeführt. Es ist zu beachten, dass der resultierende Umfang einer Scheibe in den Rahmen einer Iteration passt. Um möglichst schnell eine erste Use Case Slice für die Implementierung bereitzustellen, reicht es aus, die Story mit dem einfachsten Ablauf Basisablauf eines Use Case vorzubereiten.

In der folgenden Abbildung ist das Schneiden eines Use Case entlang seiner Abläufe visualisiert. Die daraus entstehenden Fragmente werden mit Test Cases und Storys ergänzt. Diese Fragmente (Abläufe) werden mit einer Referenz auf den zugehörigen Use Case versehen und mit dem Zustand ihrer Entwicklung behaftet, um daraus eine planbare und eigenständige Einheit (Slice) für die iterative Entwicklung zu formen.

*Diese Arbeit wurde durchgeführt bei der Firma it-Designers, Esslingen-Zell

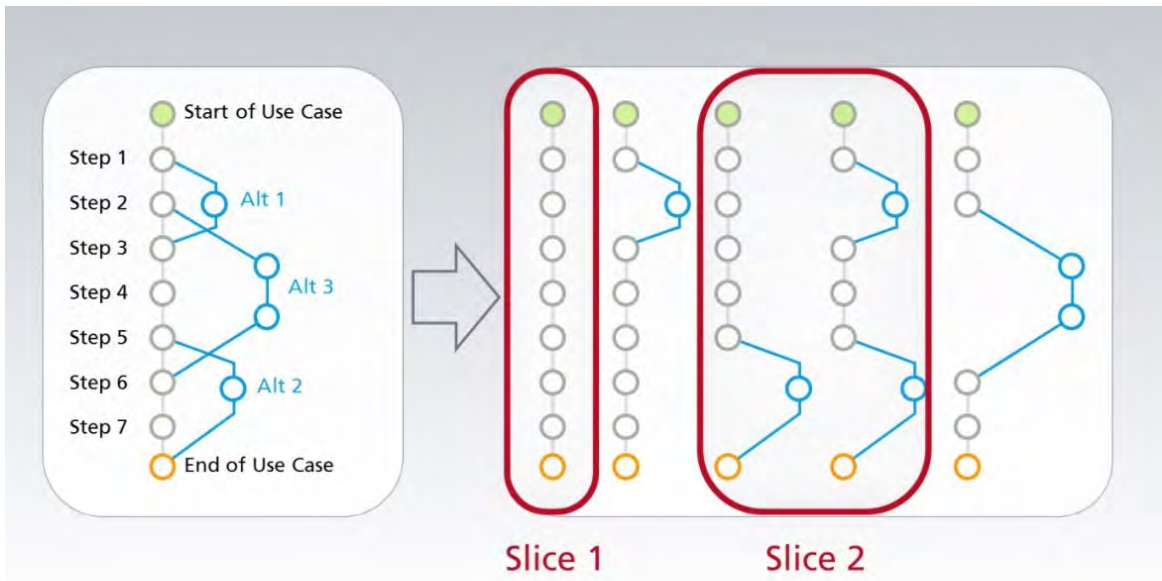


Abbildung 2: Vertikales Schneiden eines Use Case in Slices

Vergleich zu User Story Mapping

Wie schon beschrieben, ähneln sich die Ansätze von Use Case Slices und User Storys. Beide Einheiten sind in dem Zyklus einer Iteration realisierbar. Die Technik Use Case 2.0 möchte bereits von vornherein eine System Vision und eine Gesamtschau des Systems schaffen. Im Falle der Verwendung von User Storys ist User Story Mapping eine weitere Technik, welche ebenfalls ein Gesamtbild erzeugen möchte. Der zusätzliche Aufwand für die Erzeugung eines solchen Gesamtbildes ist allerdings nicht so einfach wie bei Use Case 2.0.

Bewertung

Der Hauptvorteil von Use Case Slices gegenüber User Storys liegt darin, dass die Übersicht und der Fortschritt des Gesamtsystems zu je-

der Zeit klar ersichtlich sind. Dieser Ansatz verbindet die Vorteile von spezifikationsorientierten sowie agilen Entwicklungsmethoden und ist eine sehr gute Alternative zu den bereits aus agilen Praktiken bekannten User Storys.

Ob dieser Schritt, spezifikationsorientierte Artefakte mit agilen Prozessen zu verbinden, in der heutigen Softwareentwicklung Interesse weckt und Nutzen findet, bleibt abzuwarten. Use Case 2.0 ist noch relativ jung, muss erst bekannt werden und die bereits auf andere agile Methoden eingespielten Entwickler überzeugen. Die Idee ist gut, jedoch könnte deren Marketing offensiver sein.

[1] I. Jacobson, „Use-Case 2.0 The Definitive Guide,“ Ivar Jacobson International, USA, 2011

Bildquellen:

- Abbildung 1: Eigener Entwurf, keine UML-Notation
- Abbildung 2: <http://www.microtool.de/wie-funktioniert-use-case-2-0/> Zugriff am 15.11.2015

System on a Chip für stereobasierte Kamerasysteme Hardwareentwicklung und Programmierung der Kamertreiber in VHDL

Tobias Schmid*, Werner Zimmermann, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Um die Flexibilität zu erhöhen, übernehmen Roboter in der Fertigung immer mehr Aufgaben. Dadurch entstehen aber auch mehr kritische Bereiche bei der Absicherung durch bewegte Teile im Arbeitsbereich und veränderte Platzbedingungen. Lichtgitter schränken oft den Bewegungsspielraum stark ein oder sind nur durch enorm aufwändige mechanische Absicherungen als Komplettschutz einer Anlage sicher. In solchen Fällen ist der Schutz mit einem Kamerasystem in Betracht zu ziehen. Das sichere 3D-Kamerasystem SafetyEYE von Pilz ermöglicht lückenloses dreidimensionales Überwachen von Gefahrenbereichen mit geringem Aufwand. Wie in Abbildung 1 dargestellt, können über eine einfache Anwendersoftware Warn- und Schutzräume definiert werden [1].

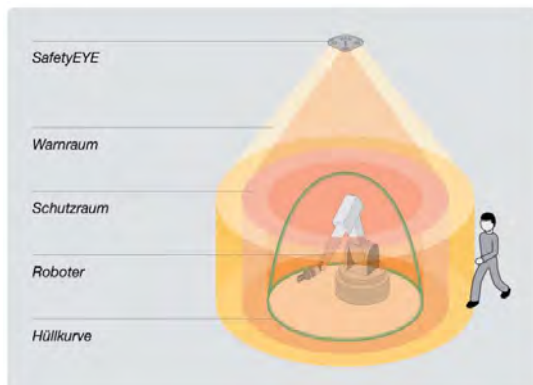


Abbildung 1: SafetyEYE Warn- und Schutzräume

Bei der Verletzung eines außenliegenden Warnraumes kann ein Signal ausgegeben werden sowie der Maschinenablauf verlangsamt werden. Dadurch ist eine Vorwarnung möglich, was bei einer Lösung mit konventionellen Lichtgittern etc. meist nicht realisierbar ist. Bei der Verletzung des Schutzraumes, welcher frei als dreidimensionaler Raum festgelegt werden kann, wird innerhalb von Sekundenbruchteilen ein NOT-Halt ausgelöst. Durch die Architektur mit drei speziellen sicherheitsgerichteten Kameras und einer zweikanaligen Auswerte- und Steuerungseinheit ist SafetyEYE SIL2 zertifiziert [2].

Die SafetyEYE Sensoreinheit sendet die Bilddaten über eine Lichtwellenleiterverbindung an die Auswerte- und Steuerungseinheit (Abb.2). Dort wird zweikanalig die komplette Bildverarbeitung und Auswertung vorgenommen und dann über eine Sicherheitssteuerung die Anlage betrieben. Wie in Abbildung 2 dargestellt, ist auch eine Ethernet-Verbindung integriert, welche zur Konfiguration des SafetyEYE Systems benötigt wird. Durch diverse Busanbindungen sind Verknüpfungen von Anlagen und Dezentralisierungen möglich [2].

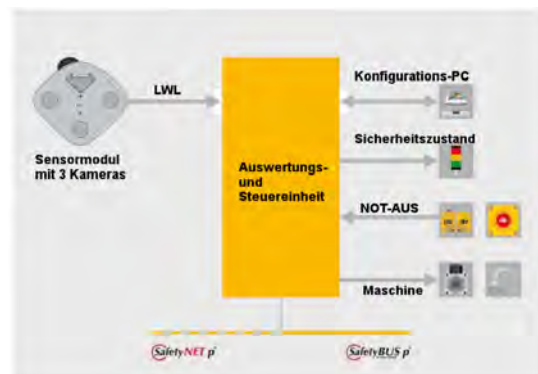


Abbildung 2: SafetyEYE Systemübersicht

Ziel der Bachelorarbeit war ein Prototyp einer neuen Sensoreinheit, mit welchem die Vorteile eines modernen System on a Chip analysiert werden können. Dies betrifft unter anderem bestimmte Bildvorverarbeitungsschritte.

Ein System on a Chip (kurz SoC) vereint mehrere Komponenten eines Systems in einem Chip. In diesem Projekt wurde ein SoC gewählt, bei dem als zentrale Komponenten zwei Prozessorkerne inklusive Cache-Speicher und eine Logikeinheit (FPGA) vorhanden sind. Zusätzlich existieren verschiedene Interfaces und Controller. Innerhalb des FPGAs lassen sich digitale Signalprozessoren erzeugen, welche auch von dem Prozessorsystem über High-Speed Interfaces genutzt werden können. Dadurch lassen sich die Vorteile von CPU, FPGA und Signalprozessoren optimal kombinieren.

*Diese Arbeit wurde durchgeführt bei der Firma Pilz GmbH & Co. KG, Ostfildern

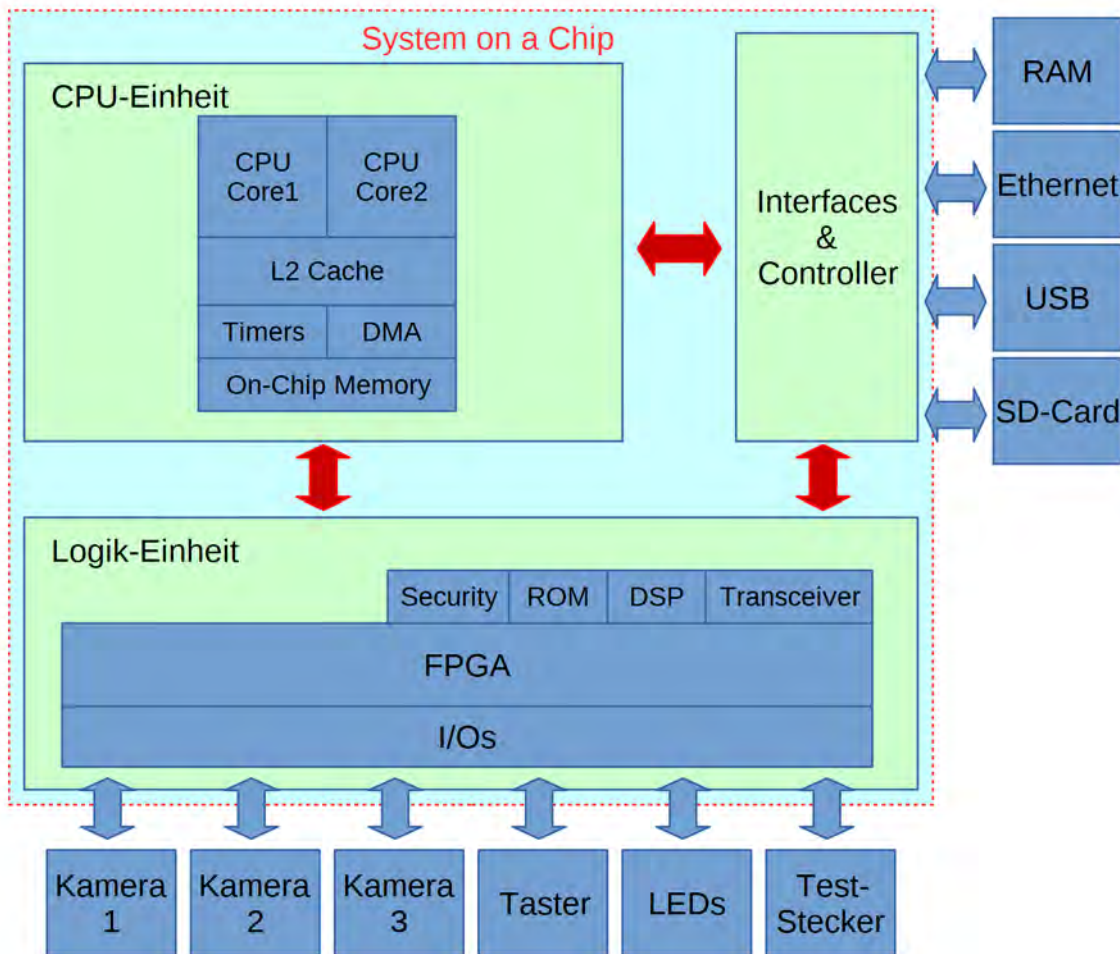


Abbildung 3: Hardware Systemübersicht

Im Rahmen der Bachelorarbeit wurde eine neue Hardwarestruktur der Sensoreinheit entwickelt, welche in Abbildung 3 verdeutlicht wird. Das Ergebnis war die komplette Entwicklung einer Platine mit Anbindung der Kameras an das System on a Chip Modul sowie einem Netzteil mit Power Sequencing. Um den Hardwareaufbau zu vereinfachen, wurde die Lichtwellenleiterverbindung zur Auswerteeinheit durch eine Kupfer Gigabit-Ethernet Verbindung ersetzt. Dadurch ist es während der Entwicklung einfacher, Bilddaten an einen Ent-

wicklungsrechner zu übertragen. Die Datenübertragung wurde auf einem embedded Linux als Serverdienst realisiert. Der Treiber für die Kameraansteuerung wurde in VHDL entwickelt und legt die Bilddaten direkt im gemeinsamen externen RAM ab. Diese Aufgabentrennung zwischen hardwareimplementierten Treibern und CPU basierenden Diensten ermöglicht zukünftig die Umsetzung komplexer Aufgabenstellungen auf kleinstem Raum.

[1] Pilz GmbH & Co. KG Produktwebseite
 [2] Pilz GmbH & Co. KG Produkthandbuch SafetyEYE

Bildquellen:

- Abbildung 1: Pilz GmbH & Co. KG
- Abbildung 2: Pilz GmbH & Co. KG
- Abbildung 3: Tobias Schmid

Web Services in einer serviceorientierten Architektur

Simon Schneider*, Joachim Goll, Manfred Dausmann

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Kommunikation zur Verknüpfung verschiedenster Arten von Systemen wie embedded oder auch mobilen Geräten gewinnt immer mehr an Bedeutung. Diese Systeme sollen über das Web miteinander kommunizieren können.

Die Idee der Kommunikation in verteilten Systemen ist jedoch nicht grundsätzlich neu. Dafür gibt es bereits die verschiedensten Technologien wie beispielsweise Remote Method Invocation (RMI), Distributed Component Object Model (DCOM) oder Common Object Request Broker Architektur (CORBA). Diese Technologien haben jedoch auch einige Nachteile. Zum einen kann es Probleme mit der Interoperabilität geben – so z. B. bei RMI. RMI ist eine Java-spezifische Umsetzung von RPC-Mechanismen. Dies bedeutet, dass RMI nicht programmiersprachenunabhängig ist. Somit kann eine Java-Anwendung nicht mit einer Anwendung, die in einer anderen Programmiersprache umgesetzt ist, via RMI kommunizieren. Zum anderen gibt es bei den zuvor genannten Technologien weitere Nachteile beispielsweise bei CORBA. In CORBA existiert zwar ein objektorientiertes, sprachübergreifendes Kommunikationskonzept, das auf verschiedene Programmiersprachen abgebildet werden kann. Jedoch benutzt CORBA proprietäre und binäre Protokolle für den Datenaustausch und hat eine starke Abhängigkeit zwischen Client und Server, womit natürlich Einschränkungen bei der Entwicklung einhergehen.

Idee

Die Idee war es nun, diese Nachteile zu umgehen. Dabei sollten Technologien, die bereits im Web etabliert sind, verwendet werden. Das Ziel war es also, dass Anwendungen, welche in verschiedenen Programmiersprachen und auf verschiedenen Plattformen entwickelt werden, dennoch über ein Netzwerk miteinander kommunizieren können. Des Weiteren werden Geschäftsprozesse oder Teile davon auf unabhängige Software-Komponenten abgebildet. Diese können in Form von Services aufgerufen werden.

Was sind Web Services?

Um diese Ziele zu erreichen, wurde von vielen

großen Unternehmen wie IBM, Microsoft und SUN die Entwicklung von Web Services gefördert. Web Services sind keine neue Technologie, sondern ein Zusammenschluss bereits bestehender und standardisierter Technologien im Web. Zu diesen bereits bestehenden Technologien zählen zum Beispiel das Hypertext Transfer Protokoll (HTTP) als Übertragungsprotokoll sowie die Metasprache Extensible Markup Language (XML) für den Informationsaustausch. Bisher gibt es noch keine standardisierte Definition des Begriffs „Web Services“. Oftmals werden Web Services als Software-Komponenten, die über ein Netzwerk in einer verteilten Umgebung mit heterogenen Systemen kommunizieren können und eine interoperable Maschine-zu-Maschine-Interaktionen unterstützen, beschrieben.

Ein Web Service kann immer über einen eindeutig zugeordneten Serviceendpoint identifiziert, erreicht und somit an diesem Endpoint aufgerufen werden. Ein Serviceendpoint ist in der Lage, Nachrichten zu empfangen, zu verarbeiten und eine Antwortnachricht zurückzusenden. Ein solcher Endpoint wird auch als Uniform Resource Identifier (URI) bezeichnet.

Ein weiteres Merkmal von Web Services ist, dass sie selbstbeschreibend sind. Dies bedeutet, dass die von einem Web Service angebotene Funktionalität, also die Nachrichten, die ausgetauscht werden können, sowie die vom Web Service verwendete Art der Übertragung, beispielsweise HTTP, in Form einer XML-basierten Schnittstellenbeschreibungssprache beschrieben wird.

Diese sogenannte Web Service Description Language (WSDL) bietet eine möglichst vollständige Beschreibung der Schnittstellen unabhängig von der konkreten Implementierung. Durch eine plattformunabhängige, XML-basierte WSDL wird zusätzlich ein hoher Grad an Interoperabilität gewonnen.

Darüber hinaus sind Web Services eine mögliche Realisierungsform für die Kommunikation in einer serviceorientierten Architektur (SOA). Weitere Realisierungsformen für eine Kommunikation in einer SOA sind beispielsweise Representational State Transfer Web Services (RESTful Web Services) oder auch CORBA.

* Diese Arbeit wurde durchgeführt bei der Firma IT-Designers GmbH, Esslingen-Zell

Serviceorientierte Architektur

Bei einer serviceorientierten Architektur bilden Services als Software-Komponenten Geschäftsprozesse oder Teile eines Geschäftsprozesses auf die Architektur eines verteilten Systems ab.

Service ist der zentrale Begriff in einer SOA. Services ergeben sich durch die Betrachtung von Geschäftsprozessen. Ein Anwendungsfall eines Geschäftsprozesses wird durch einen Service realisiert. Ein solcher Service wird daher auch als Anwendungsservice bezeichnet.

In einer serviceorientierten Architektur unterliegen Services bestimmten Anforderungen [1]. Services sind abgeschlossene Einheiten, die unabhängig voneinander aufgerufen werden können (Geschlossenheit). Außerdem stellen Services ihre Funktionalität über standardisierte Schnittstellen anderen Software-Komponenten oder Anwendungen über ein Netzwerk zur Verfügung (Verteilung). Durch eine standardisierte Schnittstellenbeschreibung sollen die Abhängigkeiten von Services untereinander beziehungsweise zwischen Services und anderen Anwendungen möglichst gering sein (lose Kopplung). Die Nutzer können Softwarekomponenten ohne Probleme austauschen, wenn sich die Schnittstellensignatur der Services nicht unterscheidet. Die Zerlegung in eine nach außen sichtbare, standardisierte Schnittstelle und die dahinter befindliche, nicht einsehbare Implementierung (Kapselung) ermöglicht eine Umsetzung auf verschiedenen Plattformen und Programmiersprachen. Des Weiteren soll ein Service keine Informationen über frühere Aufrufe haben (Zustandslosigkeit).

Ein weiteres wichtiges Ziel in einer SOA besteht darin, mehrere verfügbare Services zu kombinieren. Dafür gibt es zwei grundlegende Ansätze. Der erste Ansatz, Services zu kombinieren, ist die Orchestrierung (engl. orchestration). Bei der Orchestrierung werden mehrere, nach außen nicht sichtbare Services gekap-

selt und zu einem neuen, komplexeren Service zusammengesetzt. Diese neue Komposition eines Service repräsentiert sich nach außen als ein einziger Service. Der zweite Ansatz ist die Choreographie (engl. choreography). Dabei sollen nach außen sichtbare Services zu Geschäftsprozessen kombiniert werden. Die einzelnen verwendeten Services sind also jeweils nach außen sichtbar und werden in loser Kopplung zusammengesetzt. Als Ergebnis entsteht eine Ablaufbeschreibung, die auf die einzelnen Services zugreift [2].

Durch den Einsatz von Services entstehen die Rollen eines Servicenutzers – auch Servicekonsument –, eines Serviceanbieters und eines Serviceverzeichnisses. Die folgende Abbildung zeigt die serviceorientierte Architektur eines Web Service mit diesen drei Rollen und deren Zusammenarbeit:



Abbildung 1: Zusammenarbeit der Rollen in einer serviceorientierten Architektur

Zuerst wird ein Service von einem Serviceanbieter im Serviceverzeichnis veröffentlicht. Dabei wird unter anderem auch ein Verweis auf die Beschreibung des Service im Verzeichnis hinterlegt (1). Nachdem ein Service im Verzeichnis hinterlegt wurde, kann nun eine Suchanfrage nach einem gewünschten Service von einem Servicenutzer gestellt werden (2). Als Ergebnis der Anfrage erhält der Nutzer den Verweis auf den Service (3). Dadurch kann dieser einen Serviceaufruf beim Serviceanbieter durchführen (4) und erhält als Rückgabewert die Antwort vom Serviceanbieter (5).

[1] J. Goll und M. Dausmann, „Das Architekturmuster Service-Oriented Architecture“ in Architektur- und Entwurfsmuster der Softwaretechnik, Wiesbaden, Springer Vieweg, 2013, p.352.

[2] A. Schill und T. Springer, „Dienstorientiertes Modell“ in Verteilte Systeme, Heidelberg, Springer Vieweg, 2012, p.20

Fahrspuren- und Spurbreitenschätzung mit radarbasierten Fahrerassistenzsystemen

Natascha Schöneck*, Karlheinz Höfer, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Anzahl an Verkehrsteilnehmern ist in den letzten Jahren stetig angestiegen und wird Prognosen zu Folge auch weiterhin ansteigen [1] [2]. Um die Sicherheit jedes einzelnen Verkehrsteilnehmers zu verbessern wurden zahlreiche Fahrerassistenzsysteme entwickelt. Einige dieser Assistenzsysteme basieren auf Messdaten von Radarsensoren, so auch der Spurwechselassistent. Diese Funktion warnt den Fahrer, wenn sich ein Fahrzeug im toten Winkel befindet, oder wenn sich ein Fahrzeug mit einer höheren Geschwindigkeit auf der Nebenfahrbahn annähert. Die Warnung wird bei den meisten Automobilherstellern durch eine Leuchte im Spiegel, einem so genannten Human Machine Interface, ausgegeben.

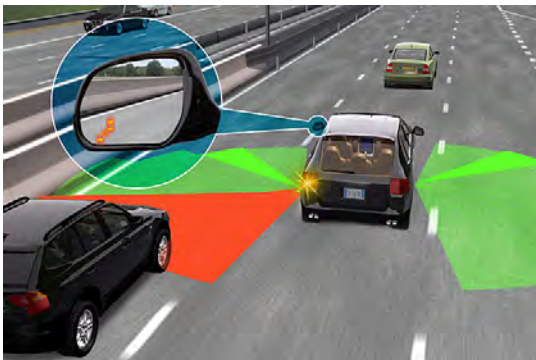


Abbildung 1: Human Machine Interface des Spurwechselassistenten

Hierbei ist es neben dem Erkennen von gefährlichen Szenarien wichtig, dass das System nur dann warnt wenn es nötig ist. Dies kann vor allem beim Spurwechsel von Vorteil sein, da sich annähernde Fahrzeuge bei bekannter Spurbreite, leichter der Fahrspur zuordnen lassen und somit ein Fehlalarm beim Fahrer verhindert werden kann. In dieser Bachelorthesis soll die technische Machbarkeit einer Spurbreitenerkennung mittels eines Radarsensors untersucht werden. Heutiger Stand der Technik im Bezug auf Spurbreitenerkennung, ist die Vermessung der Fahrbahnmarkierung durch Kamerasysteme. Für die Spurbreitenschätzung in dieser Arbeit stehen Zielmessungen eines Ra-

darsensors, der in einem Fahrzeug verbaut ist und dessen rückwärtigen und seitlichen Bereich überwacht zur Verfügung. Die vom Sensor bereit gestellten Daten beinhalten gefilterte Positions- und Geschwindigkeitsinformationen von stationären und bewegten Objekten (z.B. Verkehrsteilnehmer, Leitplanken). Aus diesen Daten lässt sich ein Histogramm der Anzahl der Zielfahrzeuge in Abhängigkeit des lateralen Abstands vom Sensor berechnen (siehe Abb. 2).

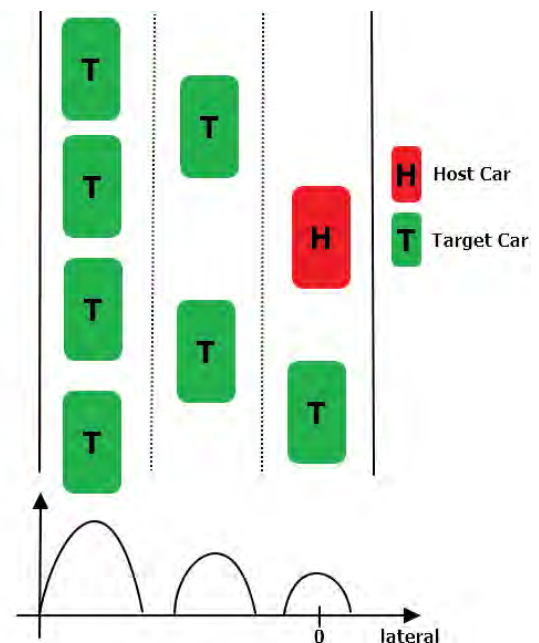


Abbildung 2: Theoretisches Fahrerszenario mit erwartetem Histogramm

Die Maxima der Histogramme sollen die Maxima ermittelt werden. Diese korrespondieren im besten Fall mit der Mitte jeder einzelnen Fahrspur. Dies lässt Rückschlüsse auf die Anzahl der Fahrspuren und deren Breite ziehen. Um diese beiden Ziele zu erreichen werden zunächst verschiedene Algorithmen aus der Clusteranalyse betrachtet. Die genaue Bewertung der einzelnen Algorithmen, wird anhand simulierter und synthetischer Daten erfolgen. Ein erstes Ergebnis eines solchen Al-

*Diese Arbeit wurde durchgeführt bei der Firma Valeo Schalter und Sensoren GmbH, Bietigheim-Bissingen

gorithmus ist in Abbildung 3 zu sehen.

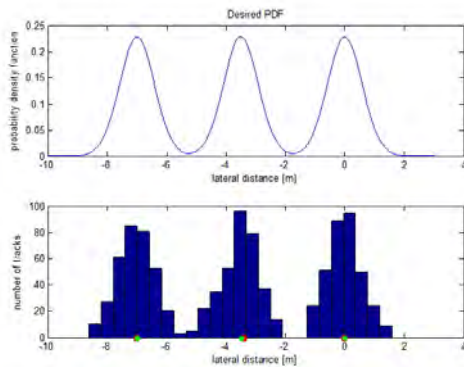


Abbildung 3: Ergebnis des Mean Shift Clustering Algorithmus

Hierbei wurden 1024 Datenpunkte aus einer multimodalen Wahrscheinlichkeitsfunktion generiert. Im Bild sind die gemessenen, anhand des getesteten Algorithmus ermittelten, lokalen Maxima mit einem roten Punkt und die tatsächlichen lokalen Maxima mit einem grünen Punkt markiert. Im Verlauf der Algorithmenanalyse zeigte sich, dass zwei Algorithmen gut geeignet sind. Diese finden durch eine gradientenbasierte Funktion die lokalen Maxima. Weitere Analysen zur Robustheit zeigten, dass eine Mindestanzahl an Zielen detektiert werden muss, um eine Zuordnungsgenauigkeit von über 90 Prozent erreichen zu können. In einem davon unabhängigen Schritt werden die Konzepte zur Spurerkennung und Spurbreitenschätzung erstellt. Diese Konzepte basieren

auf der Angabe der Spurmittelpunkte durch die getesteten Algorithmen. So wird ein Grundszenario der Clusterverteilung wie es Abbildung 4 zeigt festgelegt.

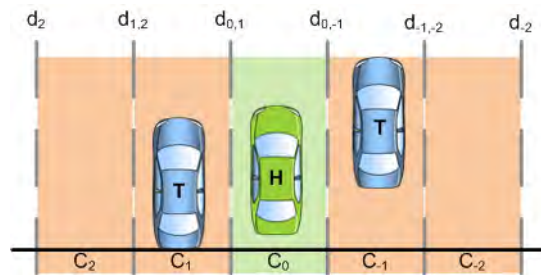


Abbildung 4: Grundszenario der Clusterverteilung

Anhand dieses Grundszenarios lassen sich Informationen zu den Fahrbahnmarkierungen, sowie zu den Spurbreiten berechnen. Sind alle Informationen zusammen getragen kann eine adäquate Aussage über die Fahrsituation, in der sich der Fahrer befindet, gemacht werden. Die Arbeit bezieht sich auf den Spurwechselassistent, welcher eine Funktion ist die in vielen Serienfahrzeugen vorhanden ist. Eine Verbesserung dieser Funktion, durch die Minimierung der Falschalarmrate, könnte zukünftig ein aktives Eingreifen des Systems ermöglichen. Dies wiederum wäre ein weiterer Schritt in Richtung autonomes Fahren. Weitere Ideen zur verbesserten Spuruweisung, wäre das einbinden von Messdaten zur Infrastruktur. Hierbei könnte der Abstand zwischen Leitplanke und Leitpfosten die gesamte Fahrbahnbreite markieren.

[1] http://www.kba.de/DE/Statistik/Fahrzeuge/Neuzulassungen/2014_n_jahresbilanz.html

Letzter Aufruf am 28.11.2015

[2] <http://de.statista.com/statistik/daten/studie/74433/umfrage/neuzulassungen-von-pkw-in-deutschland/>

Letzter Aufruf am 28.11.2015

Bildquellen:

- Abbildung 1: http://www.auto-news.de/webcore/data/content/Auto_Article_EXT/19389_1184588191320.jpg
- Abbildung 2: Eigene Abbildung
- Abbildung 3: Eigene Abbildung
- Abbildung 4: Eigene Abbildung

Entwurf und Realisierung der Diskretisierung des Simulink-Modells eines pneumatischen Linearantriebs von Festo sowie Integration des diskretisierten Modells in die Entwicklungsplattform

Jens Schuler*, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Die Produkte von Festo bestehen zu großen Teilen aus mechatronischen Systemen. Während der Platzbedarf von elektronischen Komponenten immer geringer wird, benötigt die Mechanik auch heutzutage noch vergleichsweise viel Bauraum. Die Größe der Produkte übersteigt dadurch schnell die Kapazitäten für einen schnellen und einfachen Aufbau am Arbeitsplatz, so dass spezielle Räumlichkeiten geschaffen werden müssen. Bei Produkten wie dem Linearantrieb von Festo (Abbildung 1) der nicht nur Mechanik und



Abbildung 1: Doppelwirkender Linearantrieb der Festo AG & Co. KG

Elektronik vereint, sondern auch noch ein pneumatisches Segment mit sich bringt, muss zusätzlich noch Druckluft zur Verfügung gestellt werden. Im elektrischen Bereich des Linearantriebs wird die Software implementiert. Um Funktionen frei von mechanischen oder pneumatischen Anteilen entwickeln zu können, soll ein Modell dieser Bereiche mit Hilfe der internen Entwicklungsplattform von Festo simuliert werden. Eine Simulation mechanischer und pneumatischer Bereiche hat den Vorteil, dass nicht jedem Entwickler ein Gerät zur Verfügung gestellt werden muss. Das spart nicht nur Zeit bei der Entwicklung, auch entfallen etwaige Kosten für entsprechende Räumlichkeiten. Ebenfalls sinken die Fertigungskosten, da eine geringere Anzahl an Prototypen

benötigt wird. An diesem Punkt greift die Strategie Software-in-the-Loop (SiL) an. Die Entwickler bekommen durch SiL die Möglichkeit, ihre Funktionen direkt an ihrem PC oder gar Notebook durch eine Simulation der entsprechenden Hardware zu testen. Durch diesen Schritt können Fehler und Schwachstellen der Software frühzeitig erkannt und behoben werden. Eine andere Möglichkeit zum Test neuer Funktionen ist Hardware-in-the-Loop (HiL). Allerdings ist HiL eine begrenzte und meist teure Ressource. Es

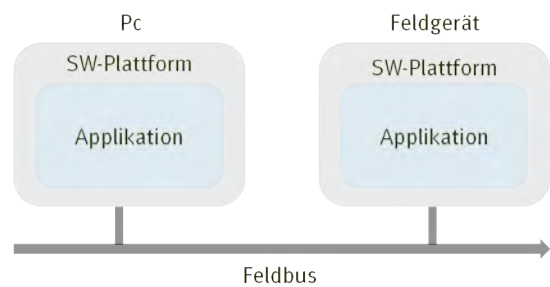


Abbildung 2: Abstrahieren der Betriebssystemebene

ist oft ein längerer Prozess vom Entwickeln der Software bis zum Test am HiL-Prüfstand. Diese Zeit entfällt bei Software-in-the-Loop vollständig [1]. Für die Kommunikation zwischen Feldgerät und Computer setzt Festo auf ein Entwicklungstool, das die Betriebssystemebene abstrahiert, so dass Software mit nur geringen Änderungen auf nahezu jedem Betriebssystem zum Einsatz kommen kann. Abbildung 2 verdeutlicht dies. Zudem wurde eine grafische Oberfläche entwickelt, um Services des Entwicklungstools zu bündeln und zur Verfügung zu stellen. Beispielsweise kann durch wenige Handgriffe eine Verbindung über einen Feldbus zwischen PC und Feldgerät aufgebaut werden, ohne dabei über genauere Kenntnisse des Protokolls zu verfügen. Zusätzlich bietet sich die Möglichkeit, während der Laufzeit dynamisch Plug-Ins zu laden und auszuführen. Die Software-Plattform von Festo hat jedoch keinen Zugriff auf Simulationen in Matlab/Simulink. Um das Modell trotz-

*Diese Arbeit wurde durchgeführt bei der Firma Festo AG & Co. KG

dem simulieren zu können und Zugriff darauf zu bekommen, soll das Modell als weiteres Plug-In in die interne Software-Plattform eingebunden werden. Hierfür wird Nutzen aus der Code-Generierung von Matlab/Simulink gezogen. Da die Plattform keine Simulationssoftware im Sinne von Matlab/Simulink ist, stellt es keine *Solver* zur Verfügung und kann somit keine numerischen Integrationsverfahren für die Darstellung kontinuierlicher Prozesse verwenden.

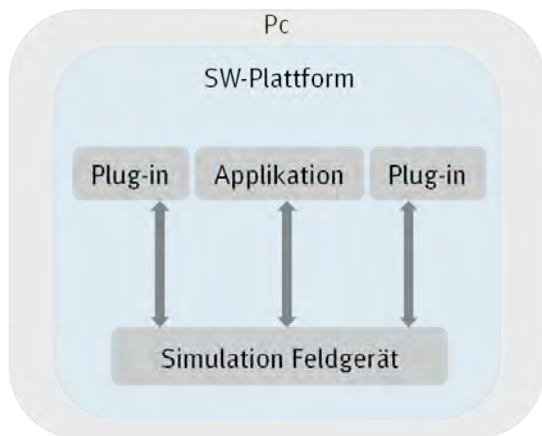


Abbildung 3: Simulation der mechanischen und pneumatischen Segmente des Linearantriebs der Festo AG & Co. KG

Das Modell muss deshalb zwingend rein diskret aufgebaut sein und darf keine zeitkontinuierlichen Funktionen beinhalten. Da durch die Modellierung der reale Ablauf eines Linearantriebs abgebildet wird, muss das Modell echtzeitfähig sein. Sind all diese Anforderungen erfüllt, wird Code aus dem Modell generiert, welcher anschließend via Plug-In in die Entwicklungsplattform von Festo eingebunden wird. Der Entwickler soll anschließend auswählen können, ob er mit dem Modell oder der Hardware arbeiten möchte. Abbildung 3 zeigt den Soll-Zustand nach Beenden der Bachel-

orthesis.

Das kontinuierliche Modell ist modular aufgebaut. Das bedeutet, dass jede Komponente des Linearantriebs einzeln als Simulink-Block implementiert ist. Bei der Diskretisierung soll dieser Vorteil nicht verloren gehen, weshalb jeder Block für sich analysiert und durch ein Subsystem mit nativen Simulink-Blöcken ersetzt wird. Eine der Herausforderungen dieser Arbeit ist das Behandeln von Diskontinuitäten. Eine Diskontinuität ist eine Unterbrechung von zeitlichen Abläufen. Der Linearantrieb von Festo besitzt durch seine minimalen und maximalen Kolbenanschläge Diskontinuitäten. Auch die Übergänge von Haftreibung in Gleitreibung stellt so eine Unterbrechung dar. Während Matlab mit einer variablen Schrittweite bei Erkennen einer Diskontinuität diese so weit reduziert, bis die Übergänge exakt getroffen werden, ist dies bei diskreten Systemen durch eine fixe Schrittweite nicht möglich. Jedoch soll das Feature bei der Diskretisierung nicht verloren gehen und so muss es auf andere Art implementiert werden. Nachdem die unterschiedlichen Komponenten übersetzt wurden, muss deren Verhalten verifiziert werden. Dazu wird jedes Modul mit seinem Original gemeinsam in einem Modell getestet. Beide Komponenten bekommen dieselben konstanten Eingangswerte. Die Ausgänge werden auf einen gemeinsamen Scope-Block gelegt. Anschließend wird das Testmodell simuliert. Durch den angeschlossenen Scope-Block kann schnell verifiziert werden, ob das Ausgangsverhalten der diskreten Komponente mit dem der kontinuierlichen übereinstimmt. Als zweiten Testlauf werden die Komponenten eingangsseitig mit realistischen Werten beschaltet. Diese umfassen sowohl Initialisierungswerte, als auch das Anfahren des maximalen und minimalen Anschlages, sowie das Umschalten der Bewegungsrichtung während einer Bewegung.

[1] QTronic GmbH: Software in the Loop Simulation <https://www.qtronic.de/de/sil.html>
Zugriffsdatum: 17.11.2015

Bildquellen:

- Abbildung 1: Festo AG & Co. KG
- Abbildung 2: Eigene Abbildung
- Abbildung 3: Eigene Abbildung

Analyse relevanter Kommunikationssysteme der Automatisierungstechnik hinsichtlich der Synchronisation mit dem unterlagerten AC-System von Festo sowie Verifikation der Ergebnisse für PROFINET, EtherCAT und Sercos III

Peter Schulze*, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Im Rahmen der Abschlussarbeit wurden die Feldbusysteme Sercos III, EtherCAT, PROFINET, PROFIBUS, EtherNet/IP und POWERLINK hinsichtlich der verschiedenen Synchronisationsmechanismen untersucht, um eine zeitlich korrekte Integration des unterlagerten AC-Systems von Festo zu ermöglichen. Diese Analyse beinhaltet die feldbus-spezifische Prozedur der Latenzermittlungen sowie Mechanismen zur Synchronisierung der Prozessdaten unterschiedlicher Feldbusse. Für simultanes Ein- und Auslesen dieser Prozessdaten in einem physisch unterlagerten System muss die Laufzeit zum AC-Master, der Referenz, sowie

die Verarbeitungszeit der Teilnehmer individuell ermittelt werden. Diese beiden Werte ergeben die Latenz und werden durch den Master berechnet, sowie im AC-System propagiert. Erreicht das Synchronisationssignal einen Teilnehmer, so ist diesem dadurch der Sendezeitpunkt bekannt. Auf den Teilnehmer mit der größten Latenz muss dann entsprechend der Differenz zur eigenen Verzögerungszeit gewartet werden. Eine deutlich präzisere Möglichkeit ist die Verwendung verteilter Uhren. Diese werden über eine Kompensation der Latenz auf die Systemzeit gestellt und je nach Protokoll zyklisch korrigiert.

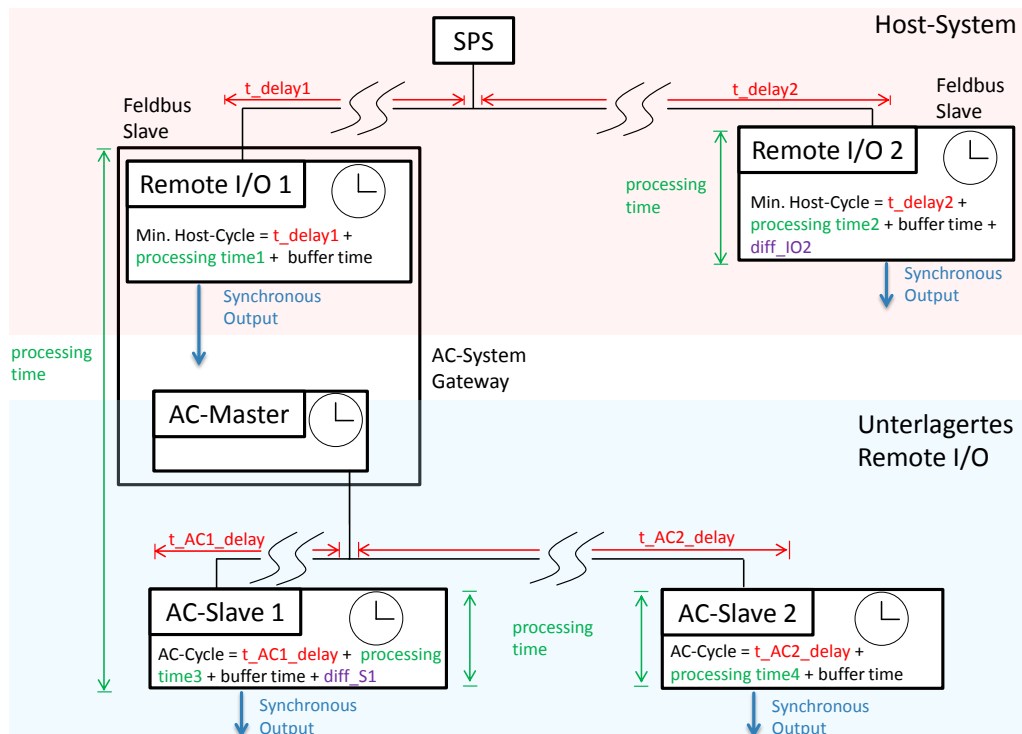


Abbildung 1: System Übersicht

Jede Uhr in dem System löst lokal ein Signal zum gleichen Zeitpunkt aus [1]. Im Wesentlichen kann die Synchronisation auf drei unterschiedliche Arten realisiert werden: *Event-*

Triggered, mit einem *Time Stamp Mechanism* oder mit einem *Distributed Clocks Mechanism*.

a) Wird auf einen Event getriggert, kann

*Diese Arbeit wurde durchgeführt bei der Firma Festo AG & Co. KG, Esslingen Zell

der daraus abgeleitete Zeitpunkt nur so exakt sein wie die Genauigkeit des Signals selbst. Im schlechtesten Fall addieren sich hier die Jitter des Hosts und des AC-Systems. Das Verfahren setzt jedoch keine verteilte Uhren voraus [2].

b) Ein *Time Stamp* (TS) kompensiert den Jitter des Eingangs-Events im überlagerten Host, sowie den Jitter innerhalb des AC-Systems, in dem der AC-Master die Sendezeitpunkte der Nachricht hinzufügt. Hierfür werden verteilte Uhren benötigt. Somit ist das Eintreffen der Daten mit

Hilfe der Zeitstempel von einem Ereignis zeitlich entkoppelt. Der Slave-Jitter des Host-Systems kann jedoch mit diesem Verfahren nicht ausgeglichen werden [1].

c) *Distributed Clocks* (DC) funktionieren wie b), jedoch wird über die Mittelwertbildung der Signal-Periode der Host-Jitter minimiert. Zusätzlich wird die Latenz eines TS vermieden, in dem das AC-System auf diese Signal-Periode angepasst wird [3].

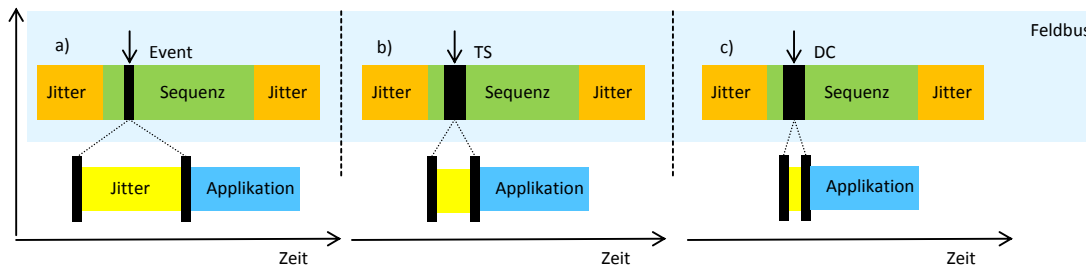


Abbildung 2: Vergleich Jitter

Des Weiteren war im Rahmen der vorliegenden Arbeit zu untersuchen, welche Informationen einem Slave des entsprechenden Feldbusses über das System, in dem er integriert ist, vorliegen. Informationen können in diesem Kontext die Zykluszeit, die Systemzeit oder der Zeitpunkt sein, welcher das Senden sowie Empfangen triggert und wie dies zu interpretieren ist. Im Rahmen der Arbeit wurde für PROFINET, EtherCAT und Sercos III jeweils ein System zur Bestätigung der Spezifikation und zur Verifikation der Datenblätterangaben aufgebaut, um Messungen zu Timing, Jitter sowie zur Synchronisationsgenauigkeit durchzuführen.

Im nächsten Schritt wurde untersucht, wie und auf welche Informationen der AC-ASIC Zugriff hat. Dies kann z.B. über einen direkten Pin, also in Hardware oder aber über das Auslesen der Software erfolgen. Hier können die aus der Analyse resultierenden Ergebnisse aller Feldbusse verglichen und in gemeinsame sowie feldbus-spezifische Anforderungen un-

terteilt werden. In Abhängigkeit des Zugriffs, in Hardware oder Software, ist über etwaige Erweiterungen und Methoden des ASIC durch z.B. zusätzliche Pins entschieden worden.

Darüber hinaus wurden verschiedene Szenarien betrachtet. Der überlagerte Feldbus ist i.d.R. langsamer als das AC-System. Dieser Fall wird unter Verwendung der minimalen Zykluszeit am häufigsten eintreten und ist durch das vielfache Senden aktueller Daten innerhalb eines überlagerten Zyklus, nach dem Prinzip des *Oversampling*, die einfachste Variante. Der letzte gültige Wert wird zum EA-Zeitpunkt des überlagerten Feldbusses übernommen.

Im letzten Teil dieser Arbeit wurde der neue Standard Time-Sensitive-Networks (TSN) nach IEEE 802.1 hinsichtlich Synchronisation untersucht. Die Analyse erfolgte insbesondere unter dem Aspekt, ob TSN neue Mechanismen zur Synchronisation bietet oder bekannte wie z.B. IEEE 1588 erweitert.

- [1] Festo AG & Co (Hrsg.): Specification AC-System. Festo Intern (Abgerufen am 15.11.2015).
 [2] EtherCAT Technology Group (Hrsg.): EtherCAT Communication. Link <https://www.ethercat.org> (Abgerufen am 09.10.2015).
 [3] Hirschmann GmbH (Hrsg.): Präzise Uhrzeitsynchronisation – IEEE 1588. Link http://www.pdv.reutlingen-universi-ty.de/rte/White_paper_ieee1588_de_v1-2.pdf (Abgerufen am 01.11.2015).

Bildquellen:

- Abbildung 1: Eigene Grafik
- Abbildung 2: Eigene Grafik

Entwicklung eines Computerspiels mit Android

Yunike Setiowati*, Reinhard Schmidt, Andreas Rößler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Heutzutage sind viele Spiele auf dem Markt erhältlich. Die Spiele sind nicht nur auf dem PC oder Laptop, sondern auch auf dem Handy oder Tablet verfügbar. Sehr viele Leute haben heutzutage ein Handy oder Tablet. Bei Wartezeiten unterwegs nutzt man für sein Lieblingsspiel oder hört seine Lieblingsmusik. Es gibt eine Vielzahl an Spielen sowohl für Android- als auch für OS-basierte mobile Geräte. Die Spiele bringen nicht nur Spaß, sie überzeugen vor allem durch eine hochwertige Grafik [1]. Die meisten App-Stores wie Google, Apple oder Amazon stellen die unterschiedlichsten Spiele, Musikrichtungen und andere funktionale Apps zur Verfügung. Es finden sich kostenlose und kostenpflichtige Apps. Die Spielarten umfassen einfache Casual-Games, Abenteuer- und Actionspiele, sowie anspruchsvolle Geschicklichkeitsspiele.

In diesem Artikel wird die Entwicklung eines Spiels für Android-basierte Geräte vorgestellt. Bei diesem Spiel namens „Reaktor“ kann die Anzahl der Spieler ausgewählt werden, so dass mit Freunden gemeinsam auf Android-Geräten gespielt werden kann. Bei diesem Spiel handelt es sich um ein Reaktionsspiel für 2 bis 4 Spieler. In Abbildung 1 ist die Auswahl der Spieleranzahl dargestellt.

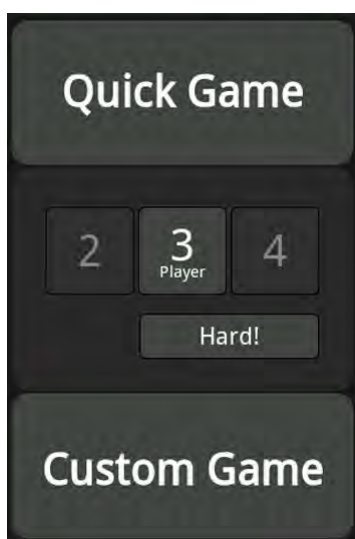


Abbildung 1: Die Auswahl von Spielanzahl

Bei diesem Spiel kann man testen, wer die bessere Reaktionsfähigkeit besitzt [2]. Wenn man auf die richtige Schaltfläche drückt, bekommt man Punkte. Im Gegensatz dazu bekommt der Gegner Punkte, wenn der falsche Knopf gedrückt wird. Während des Spiels sitzt man seinen Gegnern gegenüber, wie beispielsweise im Bus oder im Zug. Diese App hat einen geteilten Bildschirm, so dass zwei Spieler gegeneinander spielen können. Das folgende Bild 2 zeigt ein Beispiel 3 für Spieler.

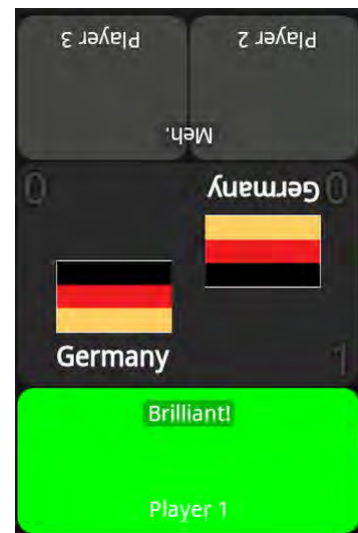


Abbildung 2: Layout von drei Spieler

Für die Entwicklung dieses Spiels wurde Android Studio verwendet. Android Studio ist eine Entwicklungsumgebung für Android-Anwendungen [3]. Die IDE zeigt die App-Layouts und vereinfacht das Generieren des Codes. Viele hilfreiche Android-Schnittstellen sind bereits integriert. Android Studio basiert auf der IntelliJ IDEA Software der Firma JetBrains. Android Studio ist kostenlos verfügbar. Es steht für die Betriebssysteme Windows, Mac OS X und Linux zur Verfügung und ersetzt die bisherige Entwicklungsumgebung Eclipse Android Development (ADT) als Hauptentwicklungsumgebung für Android Applikation von Google. Der Name Android leitet sich von der griechischen Sprache ab und bedeutet „mensenähnlich“ oder „Gestalt eines Menschen“ [4]. Der Grund, warum Android-

*Diese Arbeit wurde durchgeführt an der Fakultät Informationstechnik

Betriebssystem erstellt wurde, ist das Ziel eine einfache und intuitive Bedienung für den Anwender zu ermöglichen. Dieses Betriebssystem wird für Smartphones, Tablets, Computer und auch für Navigationssysteme genutzt. Android ist das am häufigsten verwendete mobile Betriebssystem. Es basiert auf einem lizenzfreien Linux-Kern und wurde von der Open Handset Alliance entwickelt. Die Open Handset Alliance ist eine Gruppe von 84 Technologie- und Mobilfunkunternehmen, die zusammenarbeiten, um Innovationen im mobilen Bereich zu erreichen und den Anwendern eine kostengünstige reiche und bessere mobile Anwendung anbieten zu können. Das System steht sowohl Privatpersonen als auch Unternehmen kostenlos zur Verfügung. Es kann darüber hinaus frei verändert und angepasst werden. Das folgende Bild 3 zeigt die Verteilung der weltweit genutzten Smartphone Betriebssysteme im Zeitraum von 2012 bis 2015.

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
2015Q2	82.8%	13.9%	2.6%	0.3%	0.4%
2014Q2	84.8%	11.6%	2.5%	0.5%	0.7%
2013Q2	79.8%	12.9%	3.4%	2.8%	1.2%
2012Q2	89.3%	16.8%	3.1%	4.9%	6.1%

Source: IDC, Aug 2015

Abbildung 3: Prozentsatz der weltweit genutzten Smartphone Betriebssysteme von 2012 bis 2015

Es gibt Vor- und Nachteile von Android. Einige Beispiele für Vorteile sind, dass jeder Hersteller das System anpassen und eigene Ideen umsetzen kann, Innovationen und Verbesserungen programmieren kann. Unternehmen wie HTC, Samsung, Sony und LG haben eine eigene Oberfläche über das Betriebssystem gelegt. Da Android ein offenes System ist, können vielfältig Funktionen, Erweiterungen und Modifikationen direkt auf dem System verwendet werden. Außerdem gestattet Android auch viele Änderungen an der Oberflä-

che. Ein Nachteil ist, dass verschiedenen Hersteller unterschiedliche Displaygrößen, Prozessoren, Android Versionen, Länderspezifika und andere Einschränkungen haben. Eine App wird in der Regel nur für einen Teil von Smartphones und Tablets programmiert. Aus die Displaygröße wird dabei wenig Rücksicht gelegt. Deswegen werden viele Apps nur gestreckt und vergrößert. Eine eigene Tablet-Variante gibt es nur in seltenen Fällen. Der andere Nachteil ist, dass eine längere Wartezeit für Updates und Neuerungen von Google oder anderen Herstellern besteht. Der Grund dafür ist, dass bei Software-Anpassungen anderer Hersteller der Code von Google vorab geprüft wird.

Das Spiel „Reaktor“ ist für die Studenten der Hochschule Esslingen gedacht. Die Thematik des Spiels bezieht sich auf die Hochschule Esslingen selbst. Das Ziel ist, den Studenten Wissen über ihre Hochschule spielerisch zu vermitteln. So lernen sie beispielsweise Namen der Professoren, verschiedene Fakultäten und Veranstaltungen kennen. Das Spiel soll nicht nur neuen Studenten Spaß bereiten, sondern auch Studenten in höheren Semestern können überprüfen, wie gut sie ihre Hochschule kennen. Dabei stehen verschiedene Spielmodi zur Verfügung. Fragen sind dabei entweder mit richtig oder falsch zu beantworten oder Bildern muss ein Name zugeordnet werden. Letztendlich hoffe ich, dass diese App für alle Studenten nützlich ist und Spaß macht. Diese App wurde so entwickelt, dass sie einfach weiterentwickelt und verbessert werden kann. Eine weitere Idee für ein Anwendung dieser App ist die Sicherheitsunterweisung für das Verhalten im Labor in Form von Testfragen zu implementieren. Die Studenten, die den Test bestanden haben, erhalten das Zertifikat. Damit Studenten die App schnell finden können, könnte man einen QR-Code für den Download-Adresse verwenden.

-
- [1] Die schönsten Spiele-Apps für iOS und Android, Adresse: <http://www.computerbild.de/artikel/cb-News-Handy-Apps-schoene-Spiele-Apps-iOS-Android-10055946.html>.
 - [2] Beste Android Multiplayer Spiele Apps, Mit Freunden zocken, Adresse: <http://touchportal.de/apps/beste-android-multiplayer-spiele-apps-mit-freunden-zocken/>.
 - [3] Einfachere Android-Programmierung per Android Studio , Adresse: <http://www.heise.de/developer/meldung/Einfachere-Android-Programmierung-per-Android-Studio-2-Update-1863848.html>.
 - [4] Was ist Android? Eine Einführung, Adresse: <http://www.giga.de/apps/android-os/specials/was-ist-android-eine-erklaerung/>.

Bildquellen:

- Abbildung 1:
<http://beste-apps.chip.de/android/app/2-spieler-reaktor-android-app,coolcherrytrees.games.reactor/>
- Abbildung 2:
<http://beste-apps.chip.de/android/app/2-spieler-reaktor-android-app,coolcherrytrees.games.reactor/>
- Abbildung 3:
<http://www.idc.com/prodserv/smartphone-os-market-share.jsp>

Konzeption und Realisierung eines MMI in einem verteilten System zur Erfassung und Auswertung der Emissionswerte von Fahrzeugen unter Einsatz von JavaFX und RESTful Web Services

Vyacheslav Sobin*, Manfred Dausmann, Joachim Goll

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Um die Feinstaubbelastung durch den Straßenverkehr zu senken, ist es notwendig, dass unter anderem der Kraftstoffverbrauch der Fahrzeuge verringert wird. Dem Fahrer soll geholfen werden, sein Fahrstil so zu verändern, dass er weniger Kraftstoff verbraucht. Es wird erwogen, die Verwendung schadstoffreicher Fahrzeuge erheblich einzuschränken. Dazu würde aber eine statistische Erhebung über den Schadstoffausstoß der Fahrzeuge benötigt werden.

Aus diesem Grund soll ein System erstellt werden, welches den durchschnittlichen Kraftstoffverbrauch von verschiedenen Fahrzeugtypen misst, da sich der Schadstoffausstoß proportional zum Kraftstoffverbrauch verhält [1].

Im Projekt „Sauberes Fahren“ (SF) soll mit

Hilfe der OBD-Daten [2] der durchschnittliche Schadstoffausstoß einer Autofahrt errechnet werden. Dieser soll dann zusammen mit der durchschnittlichen Geschwindigkeit an eine zentrale Leitstelle zur weiteren Auswertung gesendet werden [3].

Des Weiteren soll dem Fahrer der aktuelle Schadstoffausstoß seines Fahrzeugs auf dem Display des Fahrzeugrechners angezeigt werden, um den Fahrer durch visuelle Eindrücke zu einem umweltfreundlichen Fahren zu ermuntern. Zusätzlich soll der Fahrer diesen Wert mit dem durchschnittlichen Wert von Fahrzeugen gleichen Typs vergleichen können und den Vergleichswert auf dem Display des Fahrzeugrechners anzeigen lassen.



Abbildung 1: Schematische Darstellung des Gesamtsystems

In Abbildung 1 ist eine zentrale Leitstelle(1) zu sehen, welcher mehrere Streckenabschnitte einer Straße zugeordnet sind. Die von den Fahrzeugen produzierten Daten werden in dieser Leitstelle zusammengeführt. Als ein Verbindungsglied dient die Mobilfunkverbin-

dung des UMTS-Netz(2), welche die Fahrzeuge(3) mit der Leitstelle verbindet. Die Fahrzeuge sind mit einem UMTS-Modem ausgestattet.

*Diese Arbeit wurde durchgeführt bei der Firma IT-Desingers GmbH, Esslingen am Neckar

Die Abbildung 2 zeigt die Systemarchitektur des operationellen Systems:

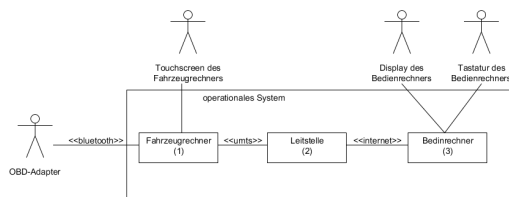


Abbildung 2: Systemarchitektur des Gesamtsystems auf der obersten Ebene

Der Fahrzeugrechner (1) verwaltet die angeschlossenen Peripheriegeräte wie das Fahrzeugdisplay mit Touchscreen und den OBD-Adapter. Der Fahrzeugrechner kommuniziert über GSM mit der Leitstelle. Das Teilsystem Leitstelle (2) dient zur persistenten Speicherung der übertragenen Fahrtdaten. Außerdem kann ein Fahrzeugrechner den durchschnittlichen Schadstoffausstoß seines Fahrzeugtyps in Abhängigkeit von seiner durchschnittlichen Geschwindigkeit abfragen. Zudem bietet die Leitstelle dem Bedienrechner Zugriffsmöglichkeiten auf alle gespeicherten Fahrtdaten. Die Komponente Bedienrechner (3) dient zur Darstellung der statistischen Auswertungen der von der Leitstelle gesammelten Fahrtdaten. Der Bedienrechner fordert die Fahrtdaten über das Ethernet bei der Leitstelle an.

In der Abschlussarbeit wurde die Konzeption und Realisierung der Software des Bedienrechners und seiner Mensch-Maschine-Schnittstelle (engl. MMI, Man-Machine-Interface) beschrieben, welche sich mit der Analyse der persistent gehaltenen Fahrtdaten aus der Datenbank der Leitstelle beschäftigt. Diese Anwendung ist auf beliebigen internetfähigen Rechnern funktionstüchtig und wird

über einen REST (Representational State Transfer) Web Service mit allen benötigten Fahrtdaten von der Leitstelle versorgt. Mit Hilfe der grafischen Oberfläche soll der Anwender in der Lage sein, den durchschnittlichen Schadstoffausstoß unterschiedlicher Fahrzeugtypen bei bestimmten Fahreigenschaften wie etwa dem Geschwindigkeitsbereich abzufragen. Die besorgten Ergebnisdaten können anschließend in unterschiedlichen Diagrammarten angezeigt und verglichen werden. Die folgende Abbildung stellt die Softwarearchitektur des Bedienrechners dar:

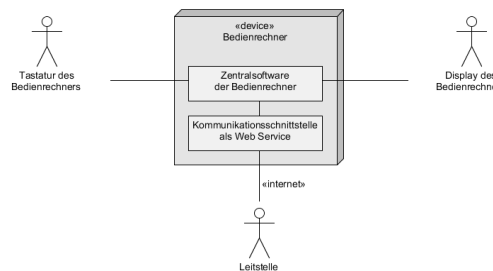


Abbildung 3: Systemarchitektur des Bedienrechners

Diese Anwendung besteht aus zwei grundlegenden Komponenten, der Zentralsoftware und der Software, die für die Kommunikation mit der Leitstelle verantwortlich ist. Die Zentralsoftware stellt die Schnittstelle zwischen dem Anwender des Bedienrechners und der Software der Leitstelle dar und ermöglicht die Visualisierung der Fahrtdaten von ausgewählten Fahrzeugen auf dem Display des Bedienrechners. Die Web Service Kommunikation dient als Schnittstelle nach außen und kommuniziert mit der Leitstelle, die die gewünschten Fahrtdaten liefert.

[1] <https://www.kfz.net/autonews/co2-verbrauch/> (Abruf am 23.11.2015)

[2] Abkürzung für On-Board Diagnose

[3] Natürlich kann dieses Verfahren problemlos noch weiter verfeinert werden

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Eigene Darstellung
- Abbildung 3: Eigene Darstellung

Analyse von Frameworks zur interaktiven Visualisierung großer Datenmengen im Web und Implementierung mit der JavaScript Bibliothek D3

Patrick Solisch*, Andreas Rößler, Reinhard Schmidt

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In der Industrie 4.0 ist die Produktion mit modernster Informations- und Kommunikationstechnik verzahnt. Durch die rasant zunehmende Digitalisierung von Wirtschaft und Gesellschaft wird diese Entwicklung weiter voranschreiten. Sie verändert nachhaltig die Art und Weise, wie in Zukunft in Deutschland produziert und gearbeitet wird. Nach Dampfmaschine, Fließband, Elektronik und IT bestimmen immer mehr nun intelligente Fabriken die vierte industrielle Revolution. [1]

1.0	1784	durch Einführung mechanischer Produktionsanlagen mithilfe von Wasser- und Dampfkraft
2.0	1870	durch Einführung arbeitsteiliger Massenproduktion durch elektrische Energie
3.0	1969	durch den Einsatz von Elektronik und IT zur weiteren Automatisierung der Produktion
4.0	Zukunft	auf Basis cyber-physischer Systeme

Abbildung 1: 4. Industrielle Revolution

So sollen möglichst alle in der Entwicklung und Produktion vorkommenden Systeme miteinander kommunizieren und interagieren, darunter sind vor allem die Menschen, Anlagen, Maschinen, Logistik und die Produkte selbst gemeint. Im Hinblick auf das Thema Industrie 4.0 sollen die Möglichkeiten und Erfahrungen festgehalten werden, ob und in wieweit es möglich ist bei einem Käufer, der beispielsweise in ein Brillengeschäft geht und schon bevor er sich eine Brille aussuchen möchte die passenden Vorschläge für ihn bereitgelegt werden. In einem späteren Schritt sollen außerdem schon Brillengestelle in einem 3D Drucker für diesen Käufer produziert werden.

Doch wie weiß ein Manager oder Chef einer Firma in welche Richtung er die Produktion seiner Produkte lenken soll? Unzählige Daten über die Produkte selbst und über die Käufer der Produkte werden gesammelt und verarbeitet. Um die richtigen Schlüsse daraus zu ziehen, müssen diese Daten gegenübergestellt werden und nur selten können große Tabellen oder einfache Grafiken den gesamten Sachverhalt der zu Grunde liegenden Daten richtig darstellen. Nun kann man die verschiedenen Betriebssysteme untersuchen und herausfinden, dass man nur durch Verwendung moderner Web Technologien nahezu unabhängig ist, welches Gerät zur Darstellung der Inhalte nötig ist. Im Web findet man häufig Grafiken bzw. Diagramme, die zwar den Sachverhalt der Daten meist gut darstellen können, diese sind jedoch nicht bzw. nur begrenzt interaktiv. Ein weiterer Nachteil dieser Form der Darstellung ist die Erstellung der Grafiken mithilfe von externen Tools, welche den passenden Output liefert. Diese müssen dann wiederum extra eingebunden werden. Darum soll ein einfaches Mittel zur interaktiven Darstellung der Daten gefunden werden.

Moderne Frameworks bieten oft die passenden Schnittstellen, um die Daten direkt aus einer Datenbank zu extrahieren und mithilfe von einfachen Funktionen passende Diagramme zu erstellen. Auch in Zukunft immer wichtiger werdender Aspekt der Darstellung im Web ist die Verwendung von "Scalable Vector Graphics" (SVG), die im Rahmen der Responsivität einen entscheidenden Vorteil gegenüber gewöhnlichen Grafiken bringen.

Im Rahmen der Bachelorarbeit sollen verschiedene Frameworks untersucht werden, die die Möglichkeiten bieten komplexe Daten möglichst einfach und intuitiv darstellen können. Zudem sollen sie die Eigenschaft haben interaktiv mit diesen Daten umzugehen. In einer ersten Analyse stellte sich heraus, dass die JavaScript Bibliothek D3 (Data-Driven-Documents) ein guter Ansatz mit einer breiten Basis an verschiedenen Möglichkeiten

*Die Arbeit wurde am Fraunhofer-Institut für Produktionstechnik und Automatisierung, Stuttgart-Vaihingen, durchgeführt.

zur Darstellung von umfangreichen Daten ist.

"Bei D3 handelt es sich um eine freie und kompakte von Mike Bostock entwickelte JavaScript Bibliothek zur effizienten Manipulation von datenbasierten Dokumenten. D3 gilt als Nachfolger von Protovis und ermöglicht die Bindung von beliebigen Daten an das Do-

document Object Model (DOM) und deren anschließende Transformation. D3 kann dabei zur Implementierung von trivialen Aufgaben, wie bspw. der Generierung einer HTML-Tabelle aus einer Datenmenge bis hin zur Implementierung von gerichteten, gewichteten Graphen als SVG verwendet werden." [2]

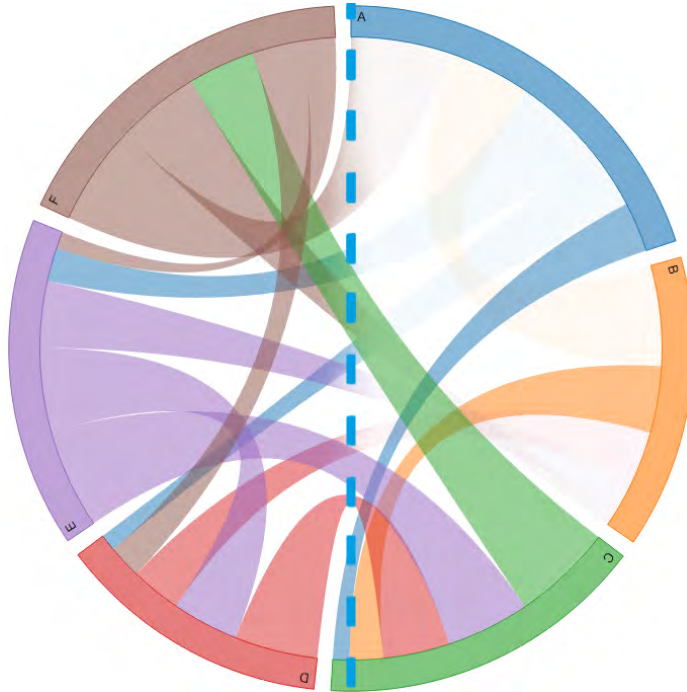


Abbildung 2: Beispieldiagramm mit der Bibliothek D3

Im weiteren Verlauf der Arbeit, sollen Szenarien entwickelt werden, wann welche Diagramme in welchem Umfang verwendet werden können und in wie weit sie eine Vorräus-

sage für mögliche Produkte gut veranschaulichen lässt.

-
- [1] Bundesministerium für Wirtschaft und Energie, 2015. Was ist Industrie 4.0? [online]. [Zugriff am: 02.12.2015] Verfügbar unter: <http://www.plattform-i40.de/I40/Navigation/DE/Industrie40/WasIndustrie40/was-ist-industrie-40.html>
- [2] Asconix Systems, 2015. Einführung in D3 [online]. [Zugriff am: 01.12.2015] Verfügbar unter: <http://www.asconix.com/howtos/javascript/d3-js-introduction>

Bildquellen:

- Abbildung 1,2: Eigene Darstellung

Konzeption und Realisierung einer Datenbank und eines Web Servers zur Kommunikation mittels RESTful Web Services in einem verteilten System zur Erfassung und Auswertung der Emissionswerte von Fahrzeugen

Tino Julian Stadelmaier*, Manfred Dausmann, Joachim Goll

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Um die Feinstaubbelastung durch den Straßenverkehr zu senken, ist es notwendig, dass unter anderem der Kraftstoffverbrauch der Fahrzeuge verringert wird. Dem Fahrer soll geholfen werden, seinen Fahrstil so zu verändern, dass er weniger Kraftstoff verbraucht. Schadstofffreie Fahrzeuge sollen identifiziert werden können.

Aus diesem Grund soll ein System erstellt werden, das den durchschnittlichen Kraftstoffverbrauch von verschiedenen Fahrzeugtypen misst, da sich der Schadstoffausstoß proportional zum Kraftstoffverbrauch verhält [1].

Im Projekt „Sauberes Fahren“ (SF) soll mithilfe der OBD-Daten[2] der durchschnittli-

che Schadstoffausstoß einer Autofahrt errechnet werden. Dieser soll dann zusammen mit der durchschnittlichen Geschwindigkeit an eine zentrale Leitstelle zur weiteren Auswertung gesendet werden.

Des Weiteren soll dem Fahrer der aktuelle Schadstoffausstoß seines Fahrzeugs auf dem Display des Fahrzeugrechners angezeigt werden, um den Fahrer durch visuelle Eindrücke zu einem umweltfreundlichen Fahren zu ermuntern. Zusätzlich soll der Fahrer diesen Wert mit dem durchschnittlichen Wert von Fahrzeugen gleichen Typs vergleichen können und den Vergleichswert auf dem Display des Fahrzeugrechners anzeigen lassen.



Abbildung 1: Schematische Darstellung des Gesamtsystems

Die Leitstelle(1) stellt das zentrale Verwaltungssystem des Projektes „Sauberes Fahren“ dar. Hier werden die vom Fahrzeugrechner(3) über das UMTS-Netz(2) übertragenen Fahrtdaten aufbereitet und persistent in einer Daten-

bank abgelegt. Des Weiteren hat ein Fahrzeugrechner die Möglichkeit alle Daten über den durchschnittlichen Schadstoffausstoß in Abhängigkeit der durchschnittlichen Geschwindigkeit, die zu dem jeweiligen Fahrzeugtyp

*Diese Arbeit wurde durchgeführt bei der Firma IT-Designers GmbH, Zell(Esslingen am Neckar)

in der Datenbank vorhanden sind, von der Leitstelle anzufordern. Diese Daten können dann als Vergleichswerte herangezogen werden, um einen Vergleich zwischen dem aktuellen Schadstoffausstoß während der Fahrt und dem Schadstoffausstoß aus der Datenbank auf dem Display des Fahrzeugrechners anzuzeigen. Zudem ist es möglich, die in der Datenbank der Leitstelle gespeicherten Daten, von einem Bedienrechner aus, zur statistischen Auswertung anzufordern, so dass dem Benutzer eines Bedienrechners eine umfangreiche Auswertung der gespeicherten Fahrtdaten auf dem Display angezeigt werden kann.

Die Anforderungen an den Rechner der Leitstelle werden mit der folgenden Softwarearchitektur umgesetzt.

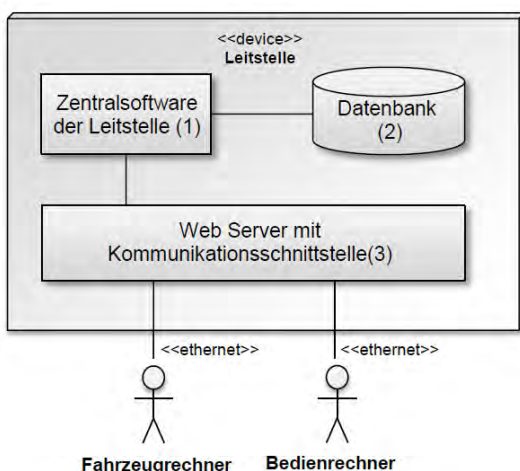


Abbildung 2: Systemarchitektur des Teilsystems Leitstelle

- **Zentralsoftware der Leitstelle (1)**

Die Zentralsoftware der Leitstelle ist für das Verwalten der Fahrtdaten zuständig. Sie bereitet die Daten aus der Datenbank in das von außen angeforderte Format auf und versendet diese über die Kommunikationsschnittstelle. Zudem kann sie die über die Kommunikationsschnittstelle eingegangenen Daten in die Datenbank weiterleiten.

- **Datenbank (2)**

Die Datenbank der Leitstelle dient als persistenter Datenhalter der Leitstelle und hält die empfangenen Fahrtdaten.

- **Web Server mit Kommunikationsschnittstelle(3)**

Dient als Schnittstelle der Leitstelle nach außen. Empfängt die Daten von Fahrzeugrechnern und Bedienrechnern und schickt ihnen angeforderte Daten.

Die Kommunikation mit den Teilsystemen Fahrzeugrechner und Bedienrechner erfolgt über eine vom Web Server zur Verfügung gestellte Kommunikationsschnittstelle. Als Zugriffsmöglichkeiten werden RESTful REpresentational State Transfer[3] Web Services implementiert, die vom Client gesendete HTTP-Requests entgegen nehmen und diese zur Verarbeitung an die Geschäftslogik weiterleiten. Nachdem die Anfragen des Client verarbeitet wurden, erhält dieser vom Web Server eine Antwort mittels HTTP-Response.

[1] <https://www.kfz.net/autonews/co2-verbrauch/> (Letzter Zugriff: 30.11.2015)

[2] Abkürzung für On-Board Diagnose

[3] https://de.wikipedia.org/wiki/Representational_State_Transfer

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Eigene Darstellung

Hardware- und Software-Entwicklung der Steuerung einer Ladesäule für Elektrofahrzeuge nach DIN EN 61851-1

Manuel Strehle*, Werner Zimmermann, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Bis 2020 soll der Treibhausgasausstoß gegenüber 1990 um 40 % reduziert werden. In Deutschland werden allein 20% der CO₂-Emissionen durch den Verkehr verursacht. Aus diesem Grund sieht die Bundesregierung in diesem Bereich ein wesentliches Einsparpotential [1]. Die Elektromobilität stellt mittel- und langfristig einen wesentlichen Baustein für einen umweltschonenden Verkehr dar. Das Ziel ist die Verwirklichung der Vision, den Strombedarf von Elektrofahrzeugen eines Tages rein aus erneuerbaren Energien zu decken, durch das gesamte Potential der Elektromobilität für den Klimaschutz [2].

Die vier „Schaufenster der Elektromobilität“ [3] dienen dem Ziel, Deutschland bis zum Jahr 2020 zum Leitmarkt für Elektromobilität zu entwickeln. Um die Elektromobilität zum Erfolg zu führen, muss eine moderne Infrastruktur zur intelligenten Vernetzung geschaffen werden, denn nur ein ansprechendes Gesamtsystem wird den Endnutzer von der Elektromobilität überzeugen. Als Teil des Gesamtsystems steht die Firmenflotte im Vordergrund dieses Regierungsprojekts. Bereits jetzt können innerhalb dieser bisher geschlossenen Flotten erhebliche Potenziale durch die Vernetzung der Ressourcen Fahrer, Fahrzeug und Infrastruktur gehoben werden. In einem weiteren Schritt können die Flotten geöffnet und durch den Einsatz von intelligenter Informationstechnik mit weiteren Elementen, z.B. weiteren Flotten und Diensten, vernetzt werden. Dieser Ansatz fördert einen experimentellen Raum, um Elektromobilität wirtschaftlich, massentauglich und skalierbar zu machen [3].

Aktuell sind einige Förderprojekte bereits abgeschlossen oder laufen gerade aus. Bei vielen geht es daher aktuell um eine Verlängerung bzw. eine Erweiterung des bisherigen Projektumfangs. In dieser Phase ist es notwendig, die erworbenen Erfahrungen zu sichten und die Weiterentwicklung des ganzen Umfelds neu zu bewerten. So gab es in den Anfängen der Elektromobilität nur einen sehr offenen Draft der DIN EN 61851, was die normativen Umsetzungen für Ladeinfrastruktur angeht. In der Zwischenzeit ist diese Norm in der dritten Fassung

überarbeitet und erweitert worden. Darüber hinaus gibt es ein neues Protokoll, welches die erweiterte Kommunikation über PLC (Power Line Communication) zulässt. Dies ist in der Norm 15118 ausgearbeitet. Damit soll die bisherige eher rudimentäre Kommunikation ersetzt werden. Des Weiteren war am Anfang für den Kunden eine einfache Bedienung wichtig, ähnlich eines normalen Tankvorgangs. In der Zwischenzeit wünschen sich sowohl Kunde als auch Betreiber der Ladeinfrastruktur einen immer aufwendigeren Funktionsumfang. So erwarten die Kunden beispielsweise die Möglichkeit, sich eine Ladesäule zu reservieren, ihren monatlichen Verbrauch genau analysieren zu können oder eine Rückmeldung des Fahrzeugs bei Abschluss oder Vollendung des Ladevorgangs zu bekommen. Zu Beginn wollten die Betreiber ihre Kunden noch an sich binden und das Laden deshalb nur an eigenen Säulen ermöglichen. Allerdings wird in der Zwischenzeit auch beim Laden von Elektrofahrzeugen das Roaming gefordert, so dass die Anzahl an Lademöglichkeiten deutlich vergrößert werden kann. Um diese Funktionalitäten überall gewährleisten zu können, ist eine internetbasierte Verwaltung mittels GSM oder LAN unabdingbar. Daher ist es an der Zeit, das bisherige Konzept vieler Ladesäulen noch einmal zu überarbeiten oder unter Umständen sogar komplett neu zu erstellen.



Abbildung 1: Ladestation mit digitaler Werbemöglichkeit

Die Arbeit soll in ihrem Umfang den bisherigen Stand bewerten und daraus ein neues Konzept erstellen, welches die aktuellen Normen und Kundenanforderungen kostengünstig

*Diese Arbeit wurde durchgeführt bei der Firma Heldele GmbH, Salach

tig und modular erfüllen soll. Im Mittelpunkt steht hierbei der Ladecontroller, durch den die Kommunikation mit dem Fahrzeug erfolgt und zusätzlich das Ladekabel verriegelt wird. Dazu wird eine Authentifizierungsmöglichkeit mittels NFC oder RFID benötigt. Hinzu kommt die Ansteuerung eines LED-Rings, um dem Benut-

zer die unterschiedlichen Zustände der Ladesäule zu signalisieren. Diese Aufgaben sollen für das Konzept auf einem Arduino Board umgesetzt werden. Die Ansteuerung der Hardware und die zeitkritischen Aufgaben werden mittels C++ umgesetzt.

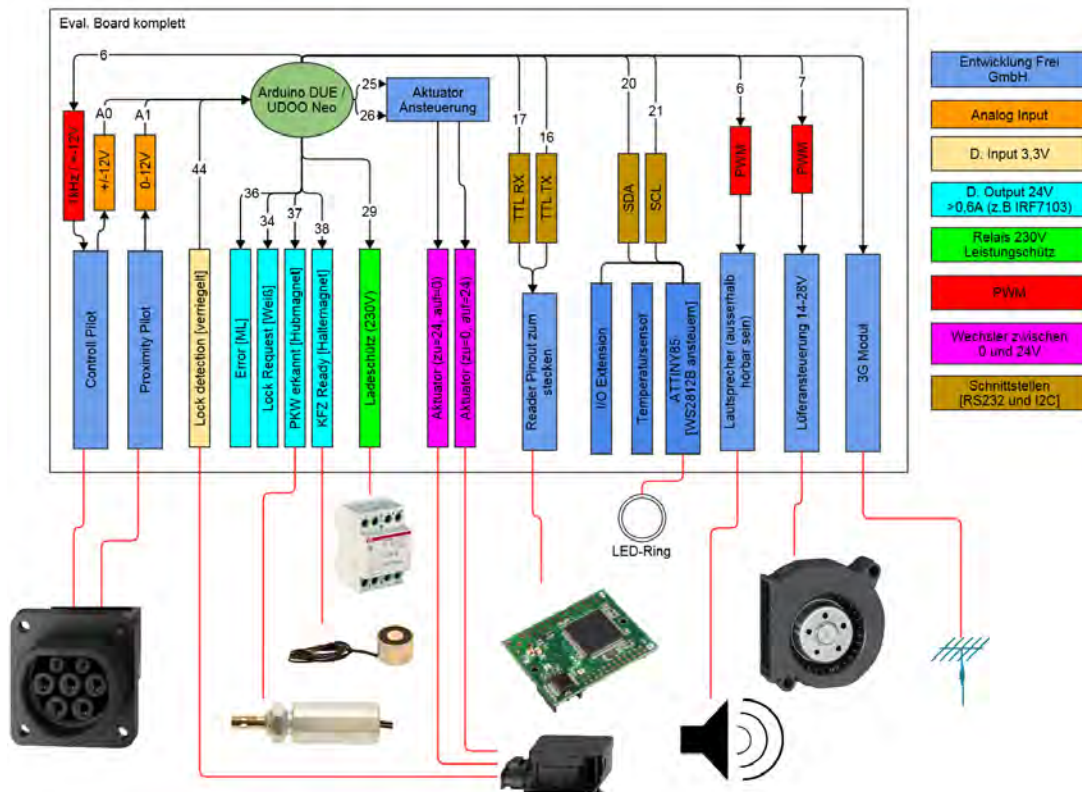


Abbildung 2: Schnittstellen der Entwicklungsplatine zur Ladesäulensteuerung

Der Zustandsautomat wird mittels Matlab/Simulink und dem Embedded Encoder in das bestehende Programm eingebunden. Dieses entwickelte Konzept soll im Anschluss an

die Arbeit zu einer Kleinserie entwickelt werden, welche dann im Feldversuch neue Erkenntnisse bringen soll.

- [1] Vgl. Centre for Co-operation with the Economies in Transition (Paris), OECD Wirtschaftsberichte 2010. 2010. (Paris: OECD, 2010), S. 82., <http://site.ebrary.com/id/10393888>.
- [2] Vgl. „Bundesregierung | Energie erzeugen | Maßnahmen im Überblick“, S.2.ff., <http://www.bundesregierung.de/Content/DE/StatischeSeiten/Breg/Energiekonzept/0-Buehne/ma%C3%9Fnahmen-im-ueberblick.html>.
- [3] Vgl. „BMW - Elektromobilität“, S.1.ff., [zugegriffen 11. November 2015, http://www.bmw.de/DE/Themen/Industrie/Industrie-und-Umwelt/elektromobilitaet,did=575166.html](http://www.bmw.de/DE/Themen/Industrie/Industrie-und-Umwelt/elektromobilitaet,did=575166.html).

Bildquellen:

- Abbildung 1: https://www.heldele.de/de/bilder/header/37_produkte_stromtankstelle.jpg?x=700
- Abbildung 2: eigene Darstellung

Konzeption und Entwicklung eines Telematik Steuergeräts auf einem Heterogenem Multiprozessorsystem

Anja Unnasch*, Harald Melcher, Andreas Rößler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Bevor man Multiprozessorsysteme entwickelt hatte, versuchten Prozessor-Hersteller die Rechenleistung durch eine immer höhere Taktrate zu steigern. Vor etwa 50 Jahren stellte Gordon E. Moore das Moore's Law auf, welches besagt, dass sich alle 24 Monate die Anzahl der Transistoren pro Fläche auf einem Mikrochip verdoppeln werde. So konnte man immer leistungsstärkere Prozessoren auch in kleinen Geräten wie Handys verbauen und auch in der Automobil-Branche konnten sich Embedded Systeme stark integrieren. Lange hat sich dieses Gesetz als wahr bewiesen, jedoch weiß man längst, dass es nicht ewig Bestand haben wird. Denn vor allem die starke Wärmeentwicklung verlangt eine Begrenzung der Taktfrequenz.

In PCs und Notebooks ist es auch deshalb mittlerweile Standard statt einem Kern mehrere Kerne auf einem System on Chip zu verbauen. So wird die große Anzahl an anstehenden Aufgaben aufgeteilt und somit schneller bewältigt. Bei der Abarbeitung von Aufgaben wird zwischen Symmetrischem Multiprocessing (MP) und Asymmetrischem MP unterschieden. Beim Symmetrischen MP haben die Kerne die gleichen Tasks abzuarbeiten und können gemeinsam eine Rechenaufgabe abarbeiten. Im Rahmen dieser Arbeit geht es um das Asymmetrische MP. Jeder Kern hat seine eigenen Aufgaben, um Multitasking zu ermöglichen. Dadurch können Programme parallel auf dem zugeordneten Kern laufen.

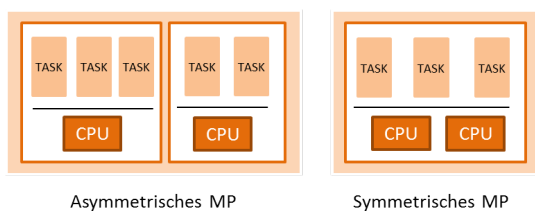


Abbildung 1: (A)symmetrisches Multiprocessing

Beim Asymmetrischen MP bietet sich eine Spezialisierung der Prozessoren an. So kann jeder Kern eine für ihn optimierte Aufgabe übernehmen. Dies nennt man Heterogenes MP. Im gebräuchlicheren Homogenen System werden

jeweils gleiche Kerne verbaut, wohingegen bei einem Heterogenen System verschiedene Kerne zum Einsatz kommen, welche spezifische Aufgaben übernehmen können. So kümmert sich ein Kern z.B. um die Signal- und Datenverarbeitung in Echtzeit, und der zweite Kern übernimmt Aufgaben für die Grafische Darstellung und den Datenspeicher.[1]

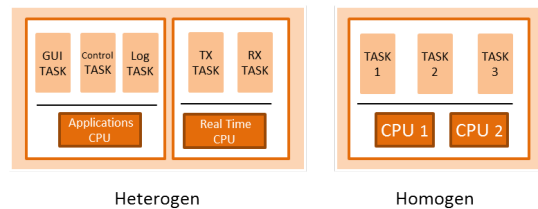


Abbildung 2: Abbildung2: eigene Entwurf

Das im Rahmen dieser Bachelorarbeit entwickelte Telematik Steuergerät läuft auf dem von Freescale Semiconductor entwickeltem i.MX6 SoloX Prozessor. Dieser Prozessor ist ein Heterogener Multiprozessor. Der erste Kern ist die ARM Cortex-A9, auf dem ein embedded Linux Betriebssystem läuft, das hauptsächlich für die grafische Benutzeroberfläche zuständig ist. Der zweite Kern ist der ARM Cortex-M4, welches für die echtzeitkritischen Aufgaben zuständig ist. Als Betriebssystem wird hier MQX verwendet, welches größtenteils von Freescale entwickelt wurde und standardgemäß auf Freescale Prozessoren mit Echtzeitfähigkeit läuft. Der Cortex-M4 hat die Aufgabe CAN-Nachrichten zu senden und zu empfangen, digitale und analoge Eingangssignale zu erfassen sowie digitale Ausgangssignale zu generieren. Außerdem liest er den auf dem i.MX6 SoloX Sabre SD Bord verbauten Beschleunigungssensor und Magnetfeldsensor aus. So gewährleistet das Board eine Echtzeitfähigkeit, obwohl rechenintensive Programme, wie die Darstellung von Grafiken, parallel laufen.

Für die Kommunikation zwischen dem Cortex-A9 und dem Cortex-M4 ist ein Shared RAM zuständig, der MCC (Multi Core Communication) genannt wird. Dafür werden über Endpunkte im Shared RAM jeweils Daten in eine Warteschlange (Queue) gelegt und auf der anderen

*Diese Arbeit wurde durchgeführt bei der Firma Gigatronik GmbH, Stuttgart

Seite nach dem FIFO-Prinzip wieder abgeholt. Adressiert werden diese Endpunkte über Core, Node, und Port.

Das embedded Linux OS auf der Cortex-A9 wurde mithilfe des Yocto Projects erstellt, welches momentan das einfachste und am besten supportete Baukastensystem für das Erstellen eines embedded Linux Betriebssystems ist. Im Grunde besteht dieser Baukasten aus Rezepten, welche die einzelnen Module herunterladen, konfigurieren, kompilieren und installieren. Diese Rezepte werden z.B. von Freescale, Open Embedded und Yocto bereitgestellt, sodass der Entwickler die für das Linux System benötigten Module leicht herausuchen und sie in den Buildprozess einbinden kann. Zudem ist es möglich ein firmenspezifisches Layer zu kreieren und dort eigene Rezepte zu schreiben, welche Treiber, Applikationen und Konfigurationen einbinden. Das damit erstellte Kernel Image hat somit schon alle Programme vorinstalliert, die auf dem Linux System laufen sollen.

Die Bereitstellung eines Qt-Layers im Yocto Project ermöglicht die Erstellung einer Toolchain für Qt, sodass ein einfaches Kompilieren und Debuggen in Linux System möglich ist. In QT wird mit C++ und mit QML programmiert. QML ist eine deklarative Programmiersprache, die es ermöglicht Benutzeroberflächen in visuellen Komponenten zusammenzustellen. Grafische und auch Animierte Oberflächen können mit Hilfe des QtQuick Moduls relativ einfach entworfen werden. Es ist zudem möglich, QML Code mit C++ Coder zu verbinden, was die Einbindung von C++ Bibliotheken ermöglicht.

Die folgende Grafik zeigt einen Überblick über die Systemarchitektur des Telematik Steuergeräts.

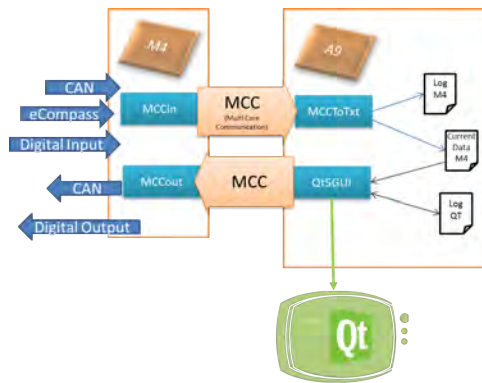


Abbildung 3: Systemarchitektur

Ankommende Nachrichten und Signale werden von dem Cortex-M4 ausgelesen und mit einem Header. Dieser Header besteht aus einem Zeitstempel, zum Zeitpunkt der Eingangszeit und einer ID, welcher Auskunft über den Inhalt der Nachricht gibt. Die Daten inclusive Header werden dann über den MCC an den Cortex-A9 weitergeleitet. Dort holt eine Applikation die Nachrichten ab und speichert die Daten in entsprechenden Dateien. Zudem läuft ein mit Qt erstelltes Programm, das für die Graphische Benutzeroberfläche zuständig ist. Auf einem angeschlossenen LCD Monitor sieht man ein Menü, an dem verschiedene Optionen ausgewählt werden können. Ein Menüpunkt ist ein Dashboard, auf dem aktuelle Daten, wie Geschwindigkeit, Motorumdrehungen, Motortemperatur und Tankfüllstand eines Automobils dargestellt werden. Da diese Daten ununterbrochen über einen CAN-Bus von der Cortex-M4 empfangen werden und im Linux System gespeichert werden, können diese Daten bei jeder Änderung aktualisiert werden. Der zweite Menüpunkt zeigt eine Klimasteuerung, bei dem Lüftung und Innentemperatur eingestellt werden können. Die geänderten Werte werden über den CAN-Bus weitergeleitet. Ein weiterer Menüpunkt zeigt die Lage und Neigung des Boards, die durch den Beschleunigungssensor ermittelt werden kann. Über ein 3D Modell des Boards kann man auf dem Bildschirm erkennen, in welcher Lage sich die Hardware gerade befindet.

[1] <https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master11-12-aw1/rempp/bericht.pdf>, Zugriff am 1.12.2015

Bildquellen:

- Abbildung 1: eigene Entwurf
- Abbildung 2: Homogenes/Heterogenes Multiprocessing
- Abbildung 3: eigene Entwurf

Entwicklung von AUTOSAR–Diagnosesoftwarekomponenten für ein Demonstrationsfahrzeug

Malte Viehweger*, Werner Zimmermann, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In der Automobilbranche gewinnen Fahrzeugdiagnosesysteme immer mehr an Bedeutung. So fallen in modernen Kraftfahrzeugen während des Betriebs verschiedene Daten an, die bei der Kfz–Inspektion zur Fehlerdiagnose ausgelesen werden können. On–Board–Diagnosesysteme (OBD) zur Einhaltung der gesetzlich vorgeschriebenen Abgasvorschriften sind bereits seit einigen Jahren für Neuzulassungen vorgeschrieben.

Zur Demonstration von Standard–Fahrzeugdiagnosefunktionen dient das „Vector Diagnostic Car“ der Firma Vector Informatik GmbH. Es handelt sich hierbei um ein Modellauto, welches ein einzelnes diagnostizierbares Automotive–Steuergerät beherbergt. Inhalt dieser Abschlussarbeit ist die Weiterentwicklung der Steuergerätesoftware zur Bereitstellung verbesserter Diagnosedienste. Das verwendete Universalsteuergerät VC121 wird von der Firma Vector produziert und eignet sich aufgrund leistungsfähiger Hardware sowie zahlreicher Schnittstellen beispielsweise als zentrales Gateway oder als I/O–Steuerung im Fahrzeugnetzwerk [1].

Über eine OBD–2–Schnittstelle kann das Vector Diagnostic Car CAN–Botschaften mit einem externen Diagnosetester austauschen. Hierbei kommt das Standardprotokoll Unified Diagnostic Services (UDS) zum Einsatz, welches einen Request–Response–Kommunikationsablauf zwischen Tester (Client) und Steuergerät (Server) ermöglicht [2]. Des weiteren beherbergt das Modellauto verschiedene LEDs (Scheinwerfer, Blinker, Brems– und Rücklichter) sowie eine Hupe, die direkt an das Steuergerät angebunden sind.

Zur Kommunikation über UDS müssen sowohl dem Tester als auch dem Steuergerät die selbe Diagnosebeschreibung bekannt sein, in der die verfügbaren Dienste, Datentypen und Datenobjekte definiert sind, welche statisch im Steuergerät implementiert sind. Durch die Beschreibung der Datentypen wird auch ein einheitliches Serialisierungsformat zur Datenübertragung geschaffen.

Im Softwarearchitekturstandard AUTOSAR wird das UDS–Protokoll durch das Basissoftwaremodul Diagnostic Communication Mana-

ger (DCM) implementiert [3]. AUTOSAR spezifiziert eine in Basissoftware und Anwendungssoftware geteilte Softwarearchitektur, durch die eine höhere Portabilität von Steuergerätesoftware im Automotive–Umfeld angestrebt wird. Während die Basissoftware in standardisierte Module unterteilt ist, haben Automobilhersteller freie Hand bei der Erstellung ihrer Anwendungssoftwarekomponenten (Software Components, SW–Cs). Zur Umsetzung bestimmter Diagnosedienste ist der DCM auf Callout–Funktionen angewiesen, die in AUTOSAR–SW–Cs implementiert werden. Der Entwurf passender SW–Cs für das Vector Diagnostic Car ist Teil dieser Abschlussarbeit.

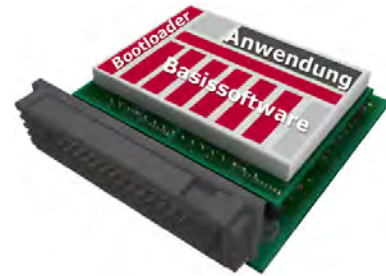


Abbildung 1: Steuergerät mit AUTOSAR–Software und Bootloader

In Abbildung 1 ist die Softwarearchitektur des Vector Diagnostic Car dargestellt. Neben der AUTOSAR–Basissoftware von Vector und den zu entwerfenden SW–Cs befindet sich auch ein sogenannter Bootloader auf dem Steuergerät, der zur Reprogrammierung der Anwendungssoftware dient. So kann der gesamte AUTOSAR–Softwarestack per UDS durch einen Diagnosetester erneuert werden.

Ein wesentliches Ziel der Arbeit war eine Erhöhung der Steuergerätfunktionalität im Vector Diagnostic Car. So hat die bisherige Embedded–Software des Demonstrators zur Ansteuerung individueller Aktoren nur den UDS–Dienst „InputOutputControlByIdentifier“ (IOCBI) bereitgestellt. Sollen beispielsweise die beiden Bremslichter eingeschaltet werden, muss der Tester zwei aufeinander folgende IOCBI–Nachrichten mit den gewünschten Ausgangswerten und den Identifiern der zu steuernden Ausgänge zum Steuergerät sen-

*Diese Arbeit wurde durchgeführt bei der Firma Vector Informatik GmbH, Stuttgart

den. Unter ausschließlicher Zuhilfenahme des IOCBI-Dienstes ist also komplexere Funktionalität wie z.B. ein Blinklicht nur durch erhöhten Aufwand im Tester zu realisieren.

Zur Umsetzung verschiedener I/O-Kontrollroutinen im Steuergerät wurde der UDS-Dienst „RoutineControl“ (RC) verwendet. Dies hat verschiedene Vorteile. Da die Funktionalität logisch im Steuergerät verankert ist,

benötigen Steuergerätestester nur noch eine Beschreibung der verfügbaren Routinen und keine zusätzliche Programmlogik. Damit wird sichergestellt, dass beliebige Diagnosetester Zugriff auf die volle Steuergerätefunktionalität haben. Nicht zuletzt sinkt durch Einsatz von RCs auch die Buslast, da lediglich UDS-Service-Aufrufe zum Starten und zum Beenden der Routinen erforderlich sind.



Abbildung 2: UDS-Kommunikationsablauf zum Starten einer Steuergeräteroutine

In Abbildung 2 ist dargestellt, wie unter Zuhilfenahme eines Diagnosetesters eine Steuergeräteroutine gestartet wird. Die eigentliche Routinenfunktionalität läuft nach dem Start in einer AUTOSAR-SW-C ab. Mit verschiedenen Steuergeräteroutinen kann nun außer der zusätzlichen Diagnosefunktionalität auch die verfügbare Hardwareplattform besser demonstriert werden.

Eine weitere Anforderung an die Vector Diagnostic Car-Steuergerätesoftware war eine verbesserte Demonstration des Steuergerätespeichers. Diagnostiziert das Steuergerät einen Fehler, so können ein im Voraus definierter Fehlercode (Diagnostic Trouble Code, DTC) und relevante Informationen zu den Fehlerumständen, die sogenannten „Snapshot-Daten“, im nichtflüchtigen Fehlerspeicher abgelegt werden. Durch eine Eindiagnosefunktion können so beispielsweise Fehler in der Ansteuerung der externen Hardwarekomponenten des Vector Diagnostic Car diagnostiziert werden.

Da jedoch reale Fehler nur sporadisch auftreten, eignet sich für Demonstrationszwecke eine Simulation von Fehlerbedingungen. Hierfür wurden verschiedene Fehler Szenarien ausgearbeitet, die sichtbare und messbare Fehler zur Folge haben. So kann der Tester nun eine Unfallsimulation herbeiführen, durch die einige Akteure scheinbar zu Schaden kommen. Die entsprechenden Ausgänge können nun nicht mehr geschaltet werden. Des Weiteren wird ein passender Fehlerspeichereintrag angelegt, der

den simulierten Fehler dokumentiert. Durch diese Vorgehensweise kann der Steuergeräte-Fehlerspeichermechanismus effektiv demonstriert werden.

Die Abschlussarbeit wurde als vollständiges Softwareentwicklungsprojekt nach in der Automobilbranche üblichen Vorgehensweisen ausgelegt. So wurde das Projekt im Rahmen einer firmeninternen Ausschreibung durch ein Lastenheft definiert, welches als Grundlage für die prozessorientierte Softwareentwicklung nach dem V-Modell diente. Es wurden somit alle Schritte von der Anforderungsanalyse über den Systementwurf und die Implementierung bis hin zur Verifikation und Validierung der Entwicklungsergebnisse durchlaufen. Insbesondere sollte durch eine vollständige Testabdeckung der Anforderungsspezifikation die Implementierung verifiziert werden. Darüber hinaus wurden Anforderungs-Design- und Testspezifikationen zur Dokumentation des Entwicklungsprozesses angefertigt, wodurch die Wiederverwendbarkeit der Software steigt.

Anhand eines exemplarischen Softwareentwicklungsprojekts konnten die On-Board-Diagnosemöglichkeiten eines modernen Fahrzeugsteuergeräts auf der Grundlage eines AUTOSAR-Systems demonstriert werden. Basierend auf dieser Arbeit können nun auch die Standard-Diagnosetools von Diagnosetestern wirkungsvoller in Szene gesetzt werden.

[1] Vector Informatik GmbH: Steuergerät VC121.

http://vector.com/portal/medien/cmc/datasheets/VC121_DataSheet_DE.pdf

[2] Werner Zimmermann, Ralf Schmidgall: Bussysteme in der Fahrzeugtechnik – Protokolle, Standards und Softwarearchitektur. Springer Vieweg, 2014, 5. Auflage, ISBN 978-3-658-02418-5

[3] AUTOSAR. <http://www.autosar.org/>

Bildquellen:

- Abbildung 1: Vector Informatik GmbH
- Abbildung 2: Malte Viehweger

Konsolidierung von AUTOSAR Steuergeräten in einer Linux Umgebung

Markus-Jonathan Wendler*, Manfred Dausmann, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Durch die stetige Zunahme der Funktionen eines Fahrzeuges wächst ebenso die Anzahl der Steuergeräte. Dies hat zur Folge, dass sich einerseits die Bauräume der Steuergeräte reduzieren und andererseits die Produktionskosten erheblich zunehmen. Die Fahrzeughersteller wünschen sich aus diesem Grund eine Optimierung in der Produktion durch die Zusammenführung mehrerer Steuergeräte auf einer Hardware.

Im Rahmen dieser Bachelorarbeit wird ein

Lösungsansatz erarbeitet, der die Konsolidierung mehrerer Steuergeräte auf einer Hardware ermöglicht. Darüber hinaus soll darauf geachtet werden, dass eine gegenseitige Beeinflussung dieser sogenannten virtuellen Steuergeräte ausgeschlossen werden kann. Des Weiteren muss untersucht werden, inwiefern die vorhandenen AUTOSAR Mechanismen angepasst werden müssen, um diese Steuergeräte-Architektur zu realisieren.

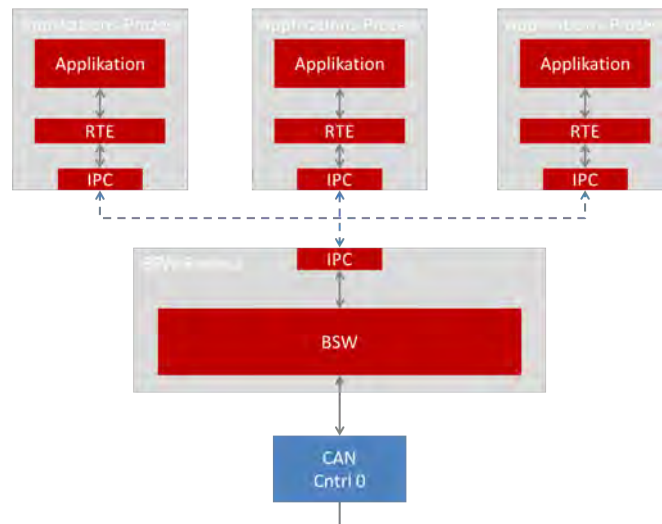


Abbildung 1: Mögliches Zielmodell zur Realisierung mehrerer unabhängiger Steuergeräte auf einer Hardware

Abbildung 1 zeigt ein mögliches Zielmodell für die Realisierung der Integration mehrerer AUTOSAR Steuergeräte auf einer Hardware. Gleichanteile der Steuergeräte, wie beispielsweise die AUTOSAR Basissoftware (BSW) werden dabei herausgezogen und sind nur einmal auf der Hardware vorhanden. Eine zusätzliche Vorgabe verhindert die Konsolidierung über den AUTOSAR Ansatz. Um dennoch eine gegenseitige Beeinflussung der einzelnen Steuergeräte ausschließen zu können, muss diese Unabhängigkeit durch einzelne Prozesse umgesetzt werden. Da ein AUTOSAR Betriebssystem keine Funktion zur Speichervirtualisierung bereitstellt, wird im Rahmen dieser Bachelorarbeit das Gesamtsystem auf einem Linux Betriebssystem ausgelegt. Anstelle des AUTOSAR OS kommt nun ein AUTOSAR OS Wrapper zum

Einsatz, sodass sämtliche Module, die eine AUTOSAR OS Schnittstelle erwarten, unverändert übernommen werden können.

Die Speichervirtualisierung ermöglicht den Ablauf der Basissoftware in einem separaten Prozess. Da die Funktionalität der Steuergeräte von der Basissoftware abhängig ist, muss ein Muster zur Interprozesskommunikation realisiert werden, das eine Kommunikation zwischen den einzelnen Parteien ermöglicht. Eine Möglichkeit wäre es, pro Applikations-Prozess zwei Pipes anzulegen, um jeweils eine Vollduplex-Kommunikation zwischen BSW und Applikation herzustellen. Um einen Prozess übergreifenden Funktionsaufruf vornehmen zu können, sind einige Umwege nötig.

*Diese Arbeit wurde durchgeführt bei der Firma Vector Informatik GmbH, Stuttgart-Weilimdorf

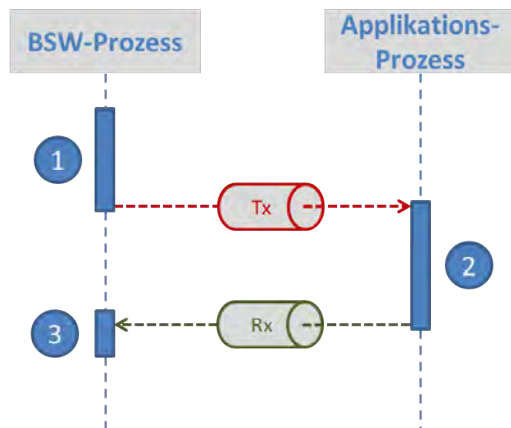


Abbildung 2: Beispiel eines Prozess übergreifenden Funktionsaufrufs

Abbildung 2 soll beispielhaft veranschaulichen, wie die BSW eine Funktion aus dem Applikationsprozess über eine Pipe aufrufen kann.

1. Unterbrechung der Funktion im BSW-Prozess, Senden der Übergabeparameter inklusive der Information, welche Funktion aufgerufen werden muss
2. Konvertierung der Übergabeparameter in den richtigen Datentyp, Aufruf der Funktion und Berechnung des Rückgabewertes
3. Empfangen des Rückgabewertes, Konvertierung des Rückgabewertes in den richtigen Datentyp und Fortführung der Funktion im BSW-Prozesses

Des Weiteren müssen die Module der Basissoftware in der Lage sein, mehrere Applikationen zu unterstützen. Dabei wird der Fokus auf die Module gelegt, die eine direkte Verbindung mit der Laufzeitumgebung (RTE) der Applikation haben. Am Beispiel des ECU State Managers (EcuM) lässt sich ein solcher Umgang mit mehreren Applikationen verdeutlichen.

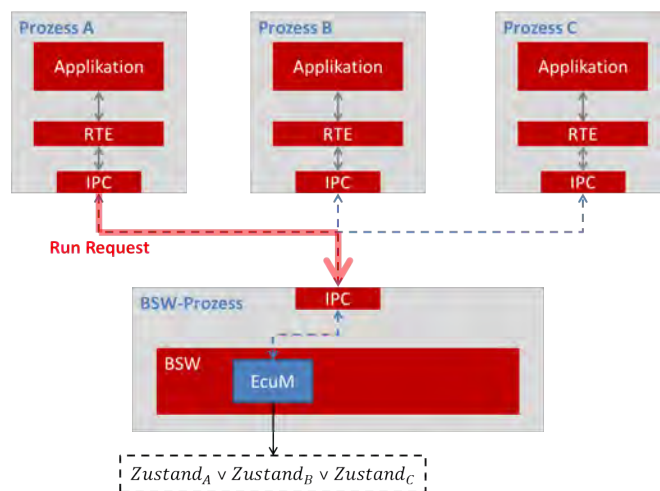


Abbildung 3: Lösungsmechanismus des EcuM-Moduls für den Multi-Applikations-Fall

Das EcuM-Modul ist für das Hoch- und Herunterfahren sowie das Aufwachen des Steuergerätes verantwortlich. [1] Es können sogenannte Run-Requests von Softwarekomponenten entgegengenommen werden, welche einen Zustandswechsel beim EcuM anfragen. Falls beispielsweise Applikation A das Steuergerät in den Zustand OFF versetzen will, muss der Zustandswechsel zwingend in Kooperation der anderen Applikationen erfolgen, da es sonst zu Betriebsstörungen des Steuergerätes kommen kann. Beim EcuM sind die einzelnen Applikationen als User registriert und somit verfügt dieses Modul über jeden Zustandsstatus. Möchte nun eine Applikati-

on durch einen Run-Request eine Zustandsänderung anfragen, so muss der EcuM eine ODER-Verknüpfung der anderen Zustandsinformationen vornehmen. Erst wenn diese im Einklang sind, kann ein Zustandswechsel des Steuergerätes vollzogen werden. (siehe Abbildung 3)

Mit Hilfe der genannten Mechanismen und anderer in ähnlicher Form soll es möglich sein, eine Zusammenführung mehrerer unabhängiger AUTOSAR Steuergeräte auf einer Hardware zu realisieren.

[1] Vector Informatik GmbH, Glossar mit Fachbegriffen und Erläuterungen zu den AUTOSAR Software Modulen (V1.8/Februar 2015)

Bildquellen:

- Abbildung 1–3: eigene Darstellung

Aufbau eines Simulationsframeworks in MATLAB für die Evaluation von Trackingalgorithmen für Parklinienmarkierungen

Carmen Wettemann*, Walter Lindermeir

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

Sicherheit und Komfort im Automobil müssen stetig wachsende Anforderungen erfüllen. Fahrzeughersteller entwickeln kontinuierlich an unterschiedlichsten Fahrerassistenzsystemen, die eine Steigerung des Fahrkomforts und eine Erhöhung der Sicherheit bewirken. Es wird von einer Vision des unfallfreien Fahrens gesprochen. Ausgehend von den heute verfügbaren Systemen soll das Auto in Zukunft ‚mitdenken‘ und den Fahrer in kritischen Verkehrssituationen unterstützen. Fahrerassistenzsysteme greifen derzeit ausschließlich teilautonom in das Fahrgeschehen ein, da aus rechtlichen und ethischen Gründen die Verantwortung der Fahrer tragen muss. Komfortfunktionen wie das autonome bzw. ferngesteuerte Einparken in eine Parklücke gewinnen zunehmend an Bedeutung. Hierfür werden Systeme mit hoch auflösenden Sensoren zur Fahrzeugumfelderfassung als auch leistungsfähige Steuergeräte entwickelt. Diese analysieren die Sensordaten in Sekundenbruchteilen und werten diese aus.

Ein Teilgebiet der Fahrerassistenzsysteme ist das Detektieren von Parklinienmarkierungen. Stand der Technik ist ein vollautonomes Einparkmanöver ohne Fahrer, sodass der Fahrer sein Fahrzeug neben einer Parklücke abstellen kann und dieses daraufhin autonom einparken lässt. Ein solches ferngesteuertes Einparkerszenario ist in Abbildung 1 dargestellt.



Abbildung 1: Ferngesteuertes Einparkerszenario

Hierfür wird für die direkte Fahrzeugumfelder-

fassung eine Nahfeldkamera mit sehr großem Öffnungswinkel verwendet. Abbildung 2 zeigt ein mögliches Einparkerszenario in eine Querparklücke aus der Sicht einer Nahfeldkamera



Abbildung 2: Einparkerszenario in eine Querparklücke aus Sicht einer Nahfeldkamera

Ein Parkliniendetektionsalgorithmus extrahiert Messpunkte auf den Parklinien im Kamerabild und übergibt sie an den Trackingalgorithmus, welcher sie über die Zeit verfolgt. Die Lage und Orientierung der Parklinie wird an den Fahrzeugbus weitergegeben, so dass das Fahrzeug autonom in die Parklücke manövriert werden kann. Dabei ist das Vermessen via Kamera einer Reihe von Störgrößen ausgesetzt. Auftretende Störgrößen sind eine falsche Kamera-Kalibrierung oder Quantisierungsfehler, die zu einer fehlerbehafteten Detektion der Parklinie führen. Um die Performance von Trackingalgorithmen isoliert bewerten zu können, sind synthetisch erzeugte Messdaten von großem Vorteil.

In dieser Bachelorarbeit wird ein Simulationsframework in MATLAB entwickelt. Dieses erzeugt synthetische Messpunkte auf ortsfesten Parklinien, während die Fahrt eines Fahrzeuges auf vorgegebenen Trajektorien simuliert wird. Die Fahrzeugtrajektorien beruhen auf vereinfachten Annahmen der Fahrdynamik. Der Berechnung liegt die einfachste Modellvorstellung der stationären Querdynamik von zweispurigen Kraftfahrzeugen, das sogenannte Einspurmodell von Riekert und Schunck von 1940, zu Grunde [1]. Dieses ist in Abbildung 3 dargestellt.

*Diese Arbeit wurde durchgeführt bei der Firma Robert Bosch GmbH, Leonberg

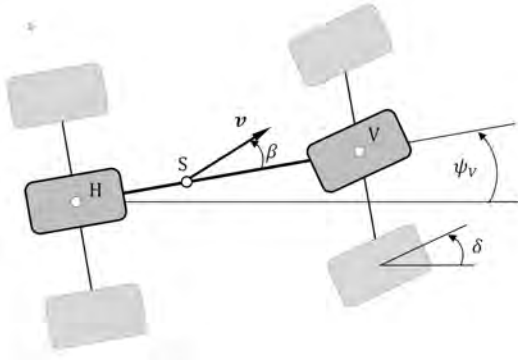


Abbildung 3: Einspurmodell nach Riekert und Schunk

Hierbei werden die Vorder- und Hinterräder achsweise durch je ein Einzelrad ersetzt. Des Weiteren werden alle Roll- und Nickbewegungen vernachlässigt. Als Bewegungsmöglichkeit verbleiben noch der Gierwinkel ψ , der Schwimmwinkel β und der Lenkwinkel δ . Der Gierwinkel kommt nur in Form der Giergeschwindigkeit vor und ist definiert durch die Drehung um die Hochachse des Fahrzeugschwerpunktes. Der Schwimmwinkel gibt die Abweichung der Schwerpunktgeschwindigkeit von der Fahrzeuglängsachse an. Die Fahrzeugbewegung beschreibt eine Kreisfahrt um einen Momentanpol bei konstanter Fahrgeschwindigkeit [2].

Für die Simulation der Fahrzeugtrajektorien wurde dieses Modell nochmals abstrahiert, so dass die momentane Geschwindigkeit und Giergeschwindigkeit die Eigenbewegung des Fahrzeuges beschreiben. Die während der Fahrt erzeugten synthetischen, feh-

lerfreien Messpunkte auf den Parklinien werden einem Trackingalgorithmus übergeben, um Lage und Orientierung der Parklinie zu berechnen. Diese Daten werden mit denen der Simulation verglichen und zur Evaluation verwendet. Die idealen Messwerte können durch verschiedene künstliche Störgrößen manipuliert werden, um reale Messungen zu simulieren. Durch diesen Vergleich soll herausgefunden werden, wie stark ein Messfehler das Detektionsergebnis verfälscht. Abbildung 3 zeigt ein simuliertes Einparkerszenario in eine Querparklücke.

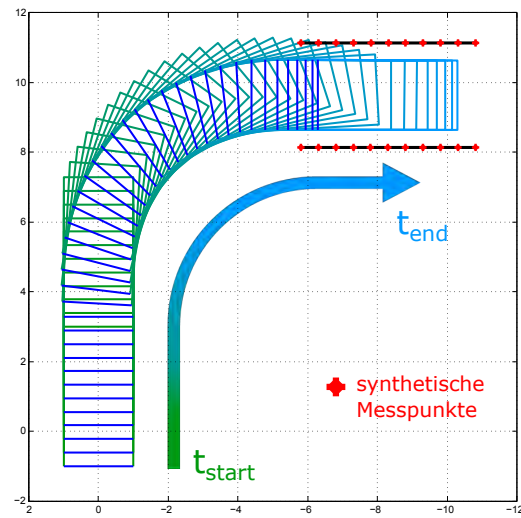


Abbildung 4: Simuliertes Einparkerszenario in eine Querparklücke

[1] Riekert / Schunk: Zur Fahrmechanik des gummibereiften Kraftfahrzeuges. Springer-Verlag, 1940.

[2] Schramm / Hiller / Bardini: Modellbildung und Simulation der Dynamik von Kraftfahrzeugen. Springer Berlin Heidelberg, 2010.

Bildquellen:

- Abbildung 1: Robert Bosch GmbH Leonberg
- Abbildung 2: Robert Bosch GmbH Leonberg
- Abbildung 3: Schramm / Hiller / Bardini: Modellbildung und Simulation der Dynamik von Kraftfahrzeugen. Springer Berlin Heidelberg, 2010.
- Abbildung 4: eigene Darstellung

Design and Implementation of an Encrypted Inter-Processor Communication for Embedded ARM Environments Using UML Based State Machine Code Generation

Marc Wöllper*, Walter Lindermeir, Reinhard Keller

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

The Harman Becker Automotive Systems GmbH “designs and engineers connected products and solutions for automakers, consumers and enterprises world wide, including connected car systems, audio and visual products, enterprise automation solutions and connected services.”[1] Today, cars are getting more and more connected on the inside and on the outside towards the surrounding environment. What does that mean? On the outside means that in the near future cars are able to communicate with each other which creates the opportunity of extended features for autonomous driving vehicles. Whereas on the inside means that electronic control units (ECUs), which implement the control functions of e.g. an engine or the infotainment system, are getting more and more complex.

Modern ECUs have more computing power than normal PCs from like 5 years ago. A modern ECU consists of several CPUs running certain tasks. For example, there are application processors, baseband processors, connectivity processors, media processors, etc. They are highly dependent to communicate with each other to provide the desired ECU functionality. As safety and security are critical areas today, ECUs have to provide these communication channels between processors being able to exchange data preferably transport encrypted. This communication channel could be run over e.g. UART, some kind of shared memory or other digital connections.

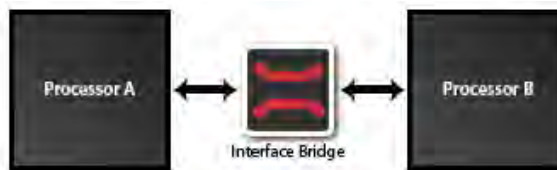


Abbildung 1: Inter-processor communication scheme

This thesis implements an encrypted communication channel for an ARM based infotainment ECU driven by a Linux operating system. The CPU runs a program which handles the whole process from serializing and deserializing data to encrypt, decrypt and transmit it via one of the existing IPC interfaces to the communication partner CPU (figure 1). There are three main classes for serialization, cryptography and interfacing. The state machine implementing the communication protocol uses the provided methods of those three classes to establish the inter-processor communication.

Serialization is done with the Google Protocol Buffers (GPB) library which is a “language-neutral, platform-neutral, extensible mechanism for serializing data – think XML, but smaller, faster and simpler”[2]. By using Google

Protocol Buffers also the structure of the transmitted data is defined in a language neutral way. The serialization happens with very small overhead, which is important especially when running on small embedded microcontrollers with only some kbytes of RAM in total.

In a second step, the serialized data gets encrypted. En- and decryption is done by the LibTomCrypt library, which provides an easy to understand application programming interface (API) for common known published block ciphers, chaining modes, pseudo-random number generators or public-key cryptography. The used encryption algorithm is Counter Mode which is a mode of operation that turns a block cipher into a stream cipher. The major advantage of a stream cipher is that the encryption or decryption process can start im-

*Diese Arbeit wurde durchgeführt bei der Firma Harman Becker Automotive Systems GmbH, Stuttgart-Vaihingen

mediately and one doesn't have to wait until there is enough data available that fits the used block size. That's why stream ciphers are particularly well suited for the use in real time systems and embedded software. After the data is serialized and encrypted successfully, it gets transmitted via a communication channel in a last step.

There are three different interfaces implemented, UART, shared memory and a file stream. Figure 2 below shows the communication process when processor A wants to send data to processor B. It can be imagined like play-

ing with a Russian Babushka. Each layer is a wooden figure which gets boxed into a larger figure until it is ready to send over the unsafe communication channel – unsafe in this case means, that it is interceptable, so a malicious attacker is able to eavesdrop the communication. After the receipt processor B decrypts and deserializes the message and processes the containing data. When processor B wants to send data to processor A it is the same procedure just vice versa.

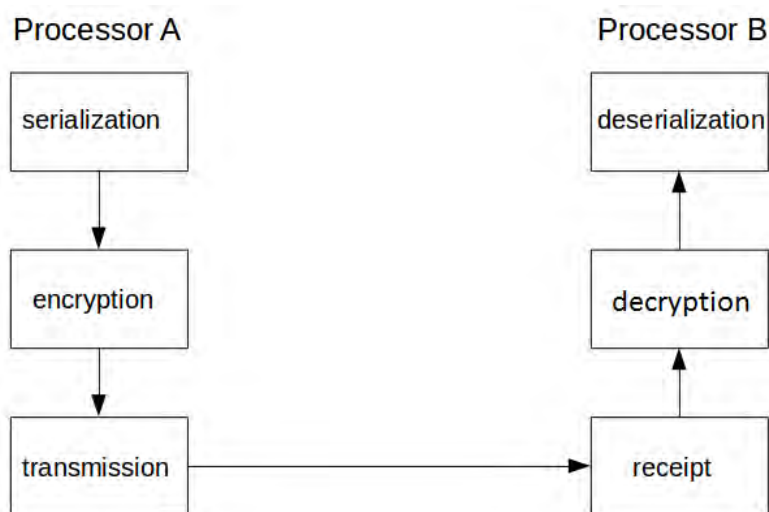


Abbildung 2: Communication model

In further steps the inter-processor communication will be extended with message authentication, error handling and a resynchronization mechanism in case of devious attacks,

occurring transmission errors or disconnections of the communication channel.

-
- [1] Harman Becker Automotive Systems GmbH – <http://www.harman.com/EN-US/pages/Home.aspx>, viewed 26.11.2015
 [2] Google Protocol Buffers – <https://developers.google.com/protocol-buffers/?hl=de>, viewed 26.11.2015
 [3] WOLF Jürgen. Linux-UNIX-Programmierung – Das umfassende Handbuch. 3. Edition. Galileo Press, 2009.
 [4] ST DENIS Tom and JOHNSON Simon. Cryptography for Developers. 1. Edition. Syngress Publishing, Inc., 2007.

Bildquellen:

- Abbildung 1: <http://www.quicklogic.com/technologies/connectivity/ipc/>, viewed 26.11.2015
- Abbildung 2: self-made

Kontinuierliche Selbstkalibrierung von Stereokameras

Christian Wolf*, Arndt Jaeger, Reiner Marchthaler

Fakultät Informationstechnik der Hochschule Esslingen – University of Applied Sciences

Wintersemester 2015/2016

In vielen verschiedenen technischen Anwendungsgebieten finden Stereokameras heutzutage immer mehr Einzug. Kontinuierlich sinkende Produktionskosten sowie die erhöhte Rechenleistung von Computern tragen zu diesem Trend erheblich bei. In Anwendungen wie beispielsweise der *Robotik* oder dem *autonomen Fahren* lässt sich dadurch kostengünstig dreidimensionale Umfelderkennung realisieren. Stereokameras bestehen aus zwei einzelnen Kameras, die über einen vorgegebenen Abstand b (die Basisweite), miteinander verbunden sind. Die mathematische Beschreibung der beiden Kameras zueinander wird über eine Translation und eine Rotation angegeben. Diese Parameter werden üblicherweise als *extrinsische* Parameter des Stereokamerasystems

bezeichnet. Als *intrinsische* Parameter werden die jeweiligen internen Kameraparameter bezeichnet, die sich aus dem Lochkamera-Modell ergeben. Das sind im Wesentlichen die zwei Brennweiten und Parameter, die die Objektivverzerrung beschreiben. Für das dreidimensionale Sehen, also das Gewinnen von geometrischen Messdaten der dreidimensionalen Welt, ist es nun erforderlich, sowohl die *intrinsischen* als auch die *extrinsischen* Parameter des Stereokamerasystems zu kennen. Dadurch lassen sich sogenannte Tiefenbilder (engl. *depth images*) erzeugen (vgl. Abbildung 1). Die Tiefe ist hierbei als Grauwert kodiert, wobei sich hellere Bildpunkte näher an der Kamera befinden als dunklere.

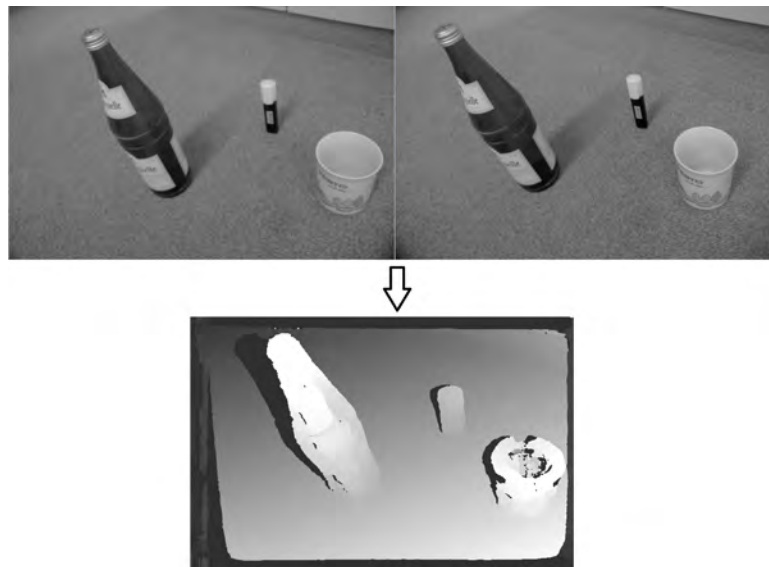


Abbildung 1: Generierung eines Tiefenbildes

Mit Hilfe des Tiefenbildes können nun dreidimensionale Weltpunkte generiert werden. Für die Ermittlung der Stereokameraparameter wird häufig beispielsweise das Kalibrierungsverfahren nach Zhang [1] angewandt. Hierbei muss ein bekanntes Referenzobjekt (oft wird hierbei ein Schachbrettmuster verwendet) in verschiedenen Perspektiven aufgenommen werden. In der Praxis muss dies *offline* durchgeführt werden und ist mit einem erheb-

lichen Zeitaufwand verbunden. Durch äußere Einflüsse können sich die kalibrierten Kameraparameter ändern, was wiederum zu einem hohen Wartungsaufwand führt. Daher wäre es wünschenswert die Kalibrierung des Stereokamerasystems *online* durchzuführen. Als Grundlage hierfür dienen die Arbeiten von Dang [2] und Balbach [3]. Hierbei beschreiben geometrische Bindungsgleichungen die nötigen Zusammenhänge durch *räumlich korre-*

*Diese Arbeit wurde durchgeführt bei der Firma Festo AG & Co. KG, Esslingen

spondierende Bildpunkte und *zeitlich korrespondierende* Bildpunkte. Mit Hilfe der geometrischen Bindungsgleichungen und den räumlich und zeitlich korrespondierenden Bildpunkten schätzt nun ein IEKF (Iterativ erweitertes Kalman-Filter) die *extrinsischen* und *intrinsischen* Parameter des Stereokamerasystems. Räumliche Punktkorrespondenzen wer-

den dabei erzeugt, indem Punktpaare aus einem Punkt im linken und einem dazu korrespondierenden im rechten Stereobild generiert werden. Dafür müssen erst sowohl im linken als auch im rechten Bild markante Punkte (engl. *Features*) gefunden werden, die möglichst einfach im korrespondierenden Stereobild wiederzufinden sind (vgl. Abbildung 2).

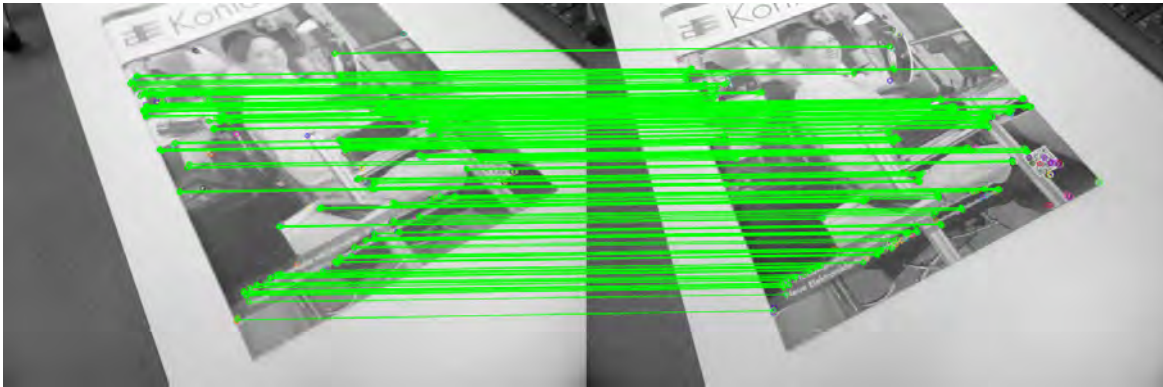


Abbildung 2: korrespondierende Punktpaare

Features werden mit einem Deskriptor beispielsweise SIFT (Scale Invariant Feature Transform) beschrieben, der nichts anderes als ein hochdimensionaler Vektor ist, der wiederum das Feature mathematisch eindeutig beschreibt. Gerade für die zeitliche Korrespondenz ist es wichtig, dass ein Feature in einer anderen Perspektive immer noch wiedererkannt werden kann. Deskriptoren müssen deshalb *rotations-* und *skalierungsinvariant*

sein, sodass Punkte über mehrere Bildverbände hinweg verfolgt werden können. Durch Verwendung dieses Verfahrens können Veränderungen am Kamerasystem (Änderungen in der Brennweite durch Neufokussierung, Änderungen der geometrischen Beziehungen der zwei Kameras durch Temperatureinflüsse) direkt in der Anwendung erkannt und korrigiert werden.

-
- [1] ZHANG,Zhengyou: A flexible new technique for camera calibration. IN: IEEE Transactions on Pattern Analysis and Machine Intelligence 22(2000), NR. 11,S. 1330–1334.
 [2] Dang Thao: Kontinuierliche Selbstkalibrierung von Stereokameras
 [3] Balbach Thilo: Automatische Selbstkalibrierung einer Stereokamera für Applikationen im Robotik-Umfeld.

Bildquellen:

- Abbildung 1: Eigene Darstellung
- Abbildung 2: Eigene Darstellung

