Eine qualitative Untersuchung zur Produktlinien-Integration über Organisationsgrenzen hinweg

Erfahrungsbericht zum Praktikum Experimentelles Software Engineering im Sommersemester 2004

Jens Kohlmeyer[1], Alexander Raschke[2], Ramin Tavakoli Kolagari[3]
[1,2]Abteilung Programmiermethodik und Compilerbau, Universität Ulm, 89069 Ulm, Germany
[3]Abteilung Software-Prozessgestaltung, DaimlerChrysler AG, Ulm
[1]jens.kohlmeyer@uni-ulm.de
[2]alexander.raschke@uni-ulm.de
[3]ramin.tavakoli_kolagari@daimlerchrysler.com

Zusammenfassung

Software-Produktlinien für eingebettete Systeme spielen eine immer wichtigere Rolle in der Industrie. Hersteller mobiler Telefone und System-Zulieferer automobiler Steuergeräte, um nur zwei Beispiele zu nennen, verwenden schon heute Software-Produktlinien als mächtige Werkzeuge, um die Qualität ihrer Prozesse und Produkte zu verbessern: Die Wiederverwendung so genannter Kernelemente wird unterstützt, womit die Entwicklungszeit verkürzt wird und die Kosten verringert werden. Die Herausforderung, der sich die Hersteller von Kraftfahrzeugen nun gegenübersehen, ist es, ihre Wiederverwendungsansätze mit den häufig bei den Zulieferern vorliegenden Produktlinien zu koppeln. Da die Automobilhersteller ihre Systeme zumeist auf der Ebene von Anforderungen spezifizieren und dabei keine abgestimmte (Werkzeug-unterstützte) Schnittstelle zu ihren Lieferanten unterhalten, um die Anforderungsspezifikationen den Hardware- und Software-Lieferanten zur Implementierung zu übergeben, kommt es zu einem Bruch in der Systematik — die Lieferanten müssen die erhaltenen Spezifikationen regelmäßig neu in ihren Produktlinien-Ansatz einbetten, was zu höheren Kosten sowie Qualitätseinbußen führt.

Das in diesem Bericht beschriebene Experiment wurde als Praktikum für Studierende im Hauptstudium an der Universität Ulm im Sommersemester 2004 durchgeführt. Wir untersuchten, inwieweit die Anwendung einer Produktlinie auf Seiten eines Automobilherstellers angepasst und mit der Wiederverwendungssystematik eines Lieferanten gekoppelt werden kann, so dass beide Seiten davon profitieren. Die aus dem Experiment gewonnenen Hinweise deuten an, dass eine sinnvolle Übertragung von Produktlinienaspekten über unterschiedliche Artefaktebenen möglich ist, ohne eine Werkzeug-unterstützte Schnittstelle zu nutzen.

1 Einleitung

Software im Bereich eingebetteter Systeme dringt in immer mehr Bereiche des Lebens vor. Software qualitativ hochwertig bei niedrigen Kosten zu erstellen, ist eine Herausforderung, die momentan Grundlage verschiedener Forschungstätigkeiten ist (beispielsweise [BFK+99, CN02, WL99]). Seit acht Jahren besteht eine enge Zusammenarbeit zwischen der Abteilung Programmiermethodik und Compilerbau der Fakultät für Informatik an der Universität Ulm und dem DaimlerChrysler Forschungszentrum in Ulm. Unter anderem werden gemeinsam betreute Praktika unter der Bezeichnung „Experimentelles Softwareengineering“ angeboten, in denen industrielle Fragestellungen hauptsächlich aus der Fahrzeugherstellung in einem experimentellen Umfeld während eines Praktikums unter kontrollierten Versuchsbedingungen untersucht werden. Auf diese Weise können für Studierende der Informatik aktuelle und interessante Fragestellungen angeboten werden, die über den sonstigen universitären Rahmen hinausgehen. Auch das DaimlerChrysler Forschungszentrum profitiert von dieser engen Zusammenarbeit, da die im Forschungszentrum erarbeiteten Herangehensweisen und Prozessverbesserungsvorschläge in einer experimentellen Untersuchung analysiert und daraufhin auf ihre Anwendbarkeit überprüft werden können. Im Sommersemester 2004 haben wir das im Folgenden beschriebene Experiment durchgeführt. Der Prozess der Automobilherstellung ist stark von der Zusammenarbeit mit Zulieferern für bestimmte Fahrzeugkomponenten abhängig. Dies gilt ebenso für die mechanischen Teile sowie für die der Software. Diese Zulieferer bieten ihre Produkte unterschiedlichen Fahrzeugherstellern an und müssen diese entsprechend an die Anforderungen der Kunden anpassen. Software-Produktlinien (kurz Produktlinien) [AHKSC01, BFK+99, CDKT01, CN02, WL99] haben sich mittlerweile auf dieser Seite aus Kosten- und Effizienzgründen etabliert. Bei dem Fahrzeughersteller werden Software-Produktlinien momentan nicht eingesetzt, da sich die Entwicklung ihrer Systeme vor allem auf die Anforderungen und der damit verbundenen Testfälle beschränkt, was die Nutzung einer Produktlinie zunächst nicht rechtfertigt, da deren Einführung mit entsprechenden Investitionskosten verbunden ist. Dennoch gehen die Automobilhersteller natürlich systematisch bei der Wiederverwendung ihrer Anforderungen vor. Allerdings unterscheidet sich die Systematik der für Anforderungen favorisierten Wiederverwendung zum Teil erheblich von dem Produktlinienvorgehen, womit es neben einem Bruch in der Entwicklung auch einen Bruch in dem Wiederverwendungsvorgehen an der Schnittstelle zwischen Auftraggeber (dem Automobilhersteller) und dem Zulieferer gibt. Die unterschiedlichen Wiederverwendungsansätze von Auftraggebern und Lieferanten lassen sich somit nicht in einen gemeinsamen Rahmen packen, was vor allem auch daran liegt, dass sich nicht nur die Systematik unterscheidet, sondern auch die Artefakte, auf die sie angewendet werden soll. Dieser Bruch in der Systematik erschwert die Planung für die Zulieferer, ist entsprechend mit höheren Entwicklungskosten verbunden, die wiederum von den Auftraggebern getragen werden müssen und stellt weiterhin Probleme für die Qualität dar, weil diese eben nicht über mehrere in der Wiederverwendungskette erstellte Systeme ganzheitlich gesichert werden kann, da der wiederverwendete Teil klein ist.

Die Nutzung von Software-Produktlinien macht für die Zulieferer Sinn, da sie im Allgemeinen sowohl die Architektur, den Code und die Softwaretests entwickeln und damit wesentliche Kostentreiber betreuen, bei denen sich ausgeklügelte Wiederverwendungsmechanismen trotz hoher Investitionskosten lohnen.

Obwohl es Ansätze gibt, wie Produktlinienideen auf Anforderungen übertragen werden können [Böc00, Fau01, HMB02, HSVM00, KS00] gibt es in der Forschung momentan kaum Ansätze, die eine Kopplung von Produktlinien über Grenzen von Organisationen hinweg diskutieren, wie es beispielsweise Sinn machen könnte für die Automobil- oder Mobiltelefonindustrie. Hier wäre die Frage zu klären, inwieweit die Nutzung von Produktliniensystematik auf Seiten der Automobilhersteller eine Software-Produktlinie auf Seiten der Zulieferer unterstützten würde. Die Forschungsfrage also, die wir in dem in diesem Bericht beschriebenen Experiment versuchten empirisch zu untersuchen, lautet: Geht die Nutzung der Produktliniensystematik auf der Ebene von Anforderungen über durchdachte Wiederverwendungsunterstützung hinaus, indem sie die auf die Anforderungserstellung folgenden Entwicklungsschritte wie Architektur- oder Codeerstellung

mit unterstützt, selbst wenn sie von einem anderen Team durchgeführt werden.

Die im Folgenden beschriebenen Ergebnisse aus dem Experiment müssen eher als empirisch belegte Hinweise, nicht als signifikante Ergebnisse im statistischen Sinne angesehen werden, da die Gruppe an teilnehmenden Studierenden klein war. In diesem Fall handelte es sich um eine Gruppe von sieben Studierenden.

Der Bericht beschreibt den Ablauf und die Auswertung des Experiments im Detail. In Abschnitt 2 geben wir eine kurze Einführung in die Grundlagen: die verwendeten Methoden und Tools wie Software-Produktlinien, das Werkzeug Statemate und die Ziele des Experiments. Abschnitt 3 illustriert den tatsächlichen Ablauf des Experiments und motiviert die wesentlichen Herausforderungen, denen wir uns als Betreuer und Forscher des Experiments gegenüber sahen. In Abschnitt 4 werden die Ergebnisse des Experiments dargestellt und objektive sowie subjektive Eindrücke diskutiert. Die Diskussion versucht die Grenzen des Experiments und dessen Designs aufzuzeigen, hinterfragt das Experiment und die Ergebnisse kritisch und diskutiert Vorschläge zu dessen Verbesserung.

2 Grundlagen und Ziele

In diesem Abschnitt werden zunächst die für das Praktikum relevanten Grundlagen der Methodik und der Werkzeuge beschrieben. Aufgrund dieser Daten werden dann die motivierenden Fragestellungen konkretisiert und die Methodik für dieses Experiment dargestellt. Für die Modellierung und die Simulation eingebetteter Systeme haben sich eine Reihe von Werkzeugen am Markt etabliert (beispielsweise MATLAB Simulink [Sim], I-Logix Statemate [Sta] oder auch Telelogic DOORS/Analyst [DOO]. In unserem Praktikum haben wir uns für das Tool Statemate (siehe auch Abschnitt 2.2) entschieden. Wir haben mit diesem Werkzeug positive Erfahrungen in vorangegangenen Praktika gemacht: Obwohl die Studenten das Tool selber anfangs nicht kennen, so kennen sie dennoch Statecharts [Har87, HN96], die die wesentliche Modellierungsmethode dieses Werkzeugs darstellen von ihrem Grundstudium, so dass die Einführungsphase in das Tool typischerweise recht kurz ist. Üblicherweise benötigen wir eine vierwöchige Einführungsphase zu Beginn des Praktikums, um neben Statemate die zu verwendenden Prozesse und Methoden zu erklären und mit Übungen zu vertiefen. Obwohl Statemate üblicherweise zur Erstellung von Spezifikationen verwendet wird, verwendeten wir das Tool zur Erstellung der finalen Produkte eines Zulieferers. Dies war insofern möglich, da eine ausführbare Semantik hinter der Syntax der Charts es ermöglichte, diese zu simulieren. Weiterhin bevorzugten wir die Möglichkeit, die zu erstellenden Systeme zu modellieren anstatt zu implementieren, um den Studierenden eine weitere Herangehensweise zur Erstellung von Systemen vorzustellen. Erfahrung aus vorherigen Praktika zeigt, dass Studierende Modellierung tatsächlich der textuellen Beschreibung von Anforderungen oder der Implementierung vorziehen.

Für unser Experiment sollten die teilnehmenden Studierenden ein Steuergerät für einen Bus entwickeln. Dies beinhaltete unter anderem die Steuerung der Türen, der Dachluken und des Fahrersitzes (näheres wird in Abschnitt 2.3 beschrieben). Für unser Experiment mussten die Studierenden nicht implementieren, es reichte, das funktionale Verhalten des Steuergeräts mit Statemate zu modellieren und zur Ausführung von Tests zu simulieren.

In den folgenden Abschnitten werden wir die Grundlagen für Produktlinien einführen. Des weiteren beschreiben wir das Tool Statemate, geben einen Überblick über den geplanten Ablauf und die Ziele des Experiments.

2.1 Software-Produktlinien

Eine *Software-Produktlinie* ist „*a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way*“ [CN02]. Die grundlegende Idee von Produktlinien ist damit die Erstellung von Kernelementen für alle bei der Entwicklung anfallenden Artefakten für einen gewissen *Scope* von Produkten, die gemeinsam verwaltet werden. Dies wird

typischerweise in einem zweiteiligen Prozess durchgeführt: *Domain Engineering* und *Application Engineering*, die beide durch Management-Aufgaben in Relation gesetzt werden. Die Konzentration auf die Kernelemente, die über alle Produkte im Bereich der Produktlinie unverändert bleiben und die Konsistenz des Ansatzes über die Artefakte hinweg, sind die beiden wesentlichen Vorteile der Produktlinie. Damit können sich die Management-Aufgaben auf die wesentlichen Aspekte (die Kernelemente) konzentrieren, ohne mit der Verwaltung beispielsweise individueller Features überladen zu sein, wie es in anderen Wiederverwendungsansätzen durchaus der Fall ist. Außerdem ist es bei der Produktentwicklung nicht notwendig, einen Pool an möglichen Features für sein System zu explorieren, wie es der Fall bei komponentenorientierten Ansätzen ist [AM02, ABM00]; stattdessen werden die Kernelemente gemäß den Bedürfnissen der anstehenden Systementwicklung erweitert. Um dies zu ermöglichen, wurde das Konzept der Variationspunkte eingeführt: Variationspunkte können zu jedem Kernelement annotiert werden, und beschreiben damit eine mögliche, angedachte Erweiterung des Systems an diesem speziellen Punkt. Des Weiteren können die Variationspunkte den Typ der Erweiterung beschreiben (beispielsweise optional, alternativ [KCH+90]) und in einem zusätzlichen Dokument können mögliche Entwurfsentscheidungen oder Abhängigkeitsbeziehungen zu anderen Variationspunkten beschrieben werden [AG01, BFK+99, HSVM00, MJA+04].

Ursprünglich in der Forschung für Software-Architekturen entwickelt [BCK03, Bos00a], finden Software-Produktlinien ihren wesentlichen Gewinn auch in diesem Bereich der Softwareentwicklung, da die Architektur der entscheidende Link zwischen den Anforderungen und dem Code ist. Änderungen an der Architektur bilden den größten Kostenfaktor bei der Softwareentwicklung. Daher ist ein ökonomischer Ansatz für die Einführung von Produktlinien nur dann sinnvoll, wenn sowohl Anforderungen als auch Architektur, Code und Testfälle adressiert werden [Coh03]. Grundsätzlich ist die Einführung von Produktlinien allerdings mit hohen Investitionskosten verbunden, im Wesentlichen bedingt durch Änderungen in der Aufbauorganisation und gesteigerten Anforderungen an die Kommunikation [BHJ+03, Bos00b, JGJ97].

2.2 Werkzeug Statemate

Für unser Experiment im Praktikum „Experimentelles Softwareengineering“ wurde das Werkzeug Statemate von I-Logix eingesetzt. Dieses Tool wird von vielen Unternehmen im Bereich der eingebetteten Systeme für den Entwurf und das Testen der herzustellenden Systeme verwendet, insbesondere auch im Automotive-Bereich. Hierarchische Zustandsautomaten (Statecharts) bilden das zentrale Modellierungselement. Sie erweitern einfache Zustandsautomaten um die Möglichkeit der Parallelisierung und der Hierarchisierung und ermöglichen dadurch eine effizientere und elegantere Modellierung, indem Zustände und Zustandsübergänge eingespart werden können. Diese Zustandsautomaten sind außerdem mit einer ausführbaren Semantik hinterlegt, so dass erstellte Diagramme simuliert werden können.

Statemate bietet dafür eine gut bedienbare Simulationsumgebung: der Benutzer hat die Möglichkeit, so genannte Panels (graphische Oberflächen) zu erstellen, mit den Statecharts zu verknüpfen und hierdurch eine prototypische Benutzeroberfläche zu erstellen. In Verbindung mit den zugrunde gelegten Statecharts bietet diese Simulationsumgebung eine hervorragende Möglichkeit, ein System zu testen und Fehler frühzeitig zu entdecken. Neben diesen Zustandsautomaten bietet Statemate sogenannte „Activitycharts“ (nicht zu verwechseln mit UML 2 Aktivitätsdiagrammen) an, um die funktionale Sicht im Systemkontext, also die Systemarchitektur, darzustellen.

Obwohl Statemate — wie oben erwähnt — normalerweise zur Erstellung von Spezifikationen eingesetzt wird, waren die erstellten Statecharts in unserem Setting außerdem bereits das fertige Produkt des Zulieferers, welches dann zum Testen an den Auftraggeber übergeben werden konnte. Mit diesem Tool ist es für die Studierenden möglich, in kurzer Zeit lauffähige Modelle zu erstellen, ohne zu viel Zeit in Programmierung und Fehlersuche zu investieren.

Ein weiterer Grund für den Einsatz dieses Tools ist, dass keiner der teilnehmenden Studierenden Erfahrungen damit hatte. Dadurch konnten die größten Unterschiede bezüglich des Vorwissens der Teilnehmer in Hinblick auf das Experiment ausgeglichen

werden. Dies verhindert auch eine übermäßige Frustration von Studenten, die mit der Programmierung im Allgemeinen und speziell von eingebetteten Systemen noch nicht so vertraut sind.

2.3 Ziele und geplante Durchführung

2.3.1 Ziele

Sieht man sich das klassische V-Modell an (siehe Abbildung 1), so fällt in Bezug auf die Konstellation Automobilhersteller–Zulieferer auf, dass der Automobilhersteller nur die „oberen“ Phasen des klassischen V-Modells (Anforderungsanalyse, Systemdesign, Integrations- und Akzeptanztest) abdeckt. Der Zulieferer dagegen ist eher bei den unteren Teilen anzusiedeln: Softwaredesign, Implementation und Modultest. Wie oben erwähnt, benutzt im Allgemeinen nur der Zulieferer Produktlinien, um die Effizienz bei der Erstellung eines Produkts zu steigern.

Interessant war für uns nun die Frage, ob sich eine Produktlinie auf Anforderungsebene, also auf den oberen Ebenen des V-Modells, für den Automobilhersteller lohnen könnte. Um diese Frage beantworten zu können, muss zunächst abgeklärt werden, ob Auftraggeber und Zulieferer, die beide eine unterschiedliche Produktlinie entwickelt haben, über diese Produktlinien kommunizieren können. Hierbei ist hauptsächlich die Kommunikation im V-Modell nach unten gemeint, also ob der Auftraggeber dem Zulieferer seine Anforderungen unter Verwendung der eigenen Produktlinie verständlich machen kann und folglich eine Art „Matching“ zwischen den beiden unterschiedlichen Produktlinien möglich ist. Der Zulieferer wird seine Produktlinie nicht offen legen, weswegen wir nur die Kommunikation wie eben beschrieben und nicht andersrum untersuchen wollten.

Neben der Untersuchung der Kommunikation über verschiedene Produktlinien hat uns auch die Frage interessiert, in wie weit die Studierenden eine Produktlinie vollständig gestalten können. Mit vollständig meinen wir in diesem Zusammenhang, wie viele Anforderungen an ein konkretes Produkt in der Produktlinie bereits als Variationspunkt vorgesehen und aufgrund dessen sehr rasch umsetzbar sind.

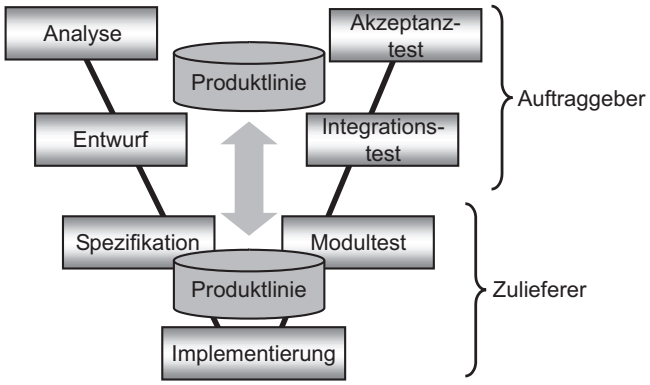


Abbildung 1: Überblick über das V-Modell im Praktikum

Schließlich wollten wir noch die zwei (wie oben beschrieben) lose gekoppelten Produktlinien mit einer kompletten Produktlinie über alle Ebenen des V-Modells hinweg miteinander vergleichen. Wieviel Mehraufwand bedeutet es, zwei voneinander unabhängige Produktlinien zu erstellen und zu pflegen? Welche Unterschiede würden bezüglich der Qualität und Wiederverwendbarkeit der Anforderungs-, Architektur- und Implementierungsdokumente festzustellen sein?

Zusammenfassend haben wir folgende Fragen mit diesem Experiment behandelt:

1. Passen die Variationspunkte der Produktlinie auf Anforderungsebene auf die Variationspunkte der Architektur- und Codeproduktlinie, obwohl diese auf den Szenarien von zwei unterschiedlichen Sichtweisen auf die Zukunft der Kernbeschreibung eines Systems herrühren?

2. In wie weit verhalten sich auf diese Art und Weise lose gekoppelten Produktlinien wie eine umfassende bezüglich der Qualität der erstellten Artefakte, Wiederverwendbarkeit der Kernpunkte und Verbrauch der Ressourcen wie z.B. Entwicklungszeit, Kommunikation und Wiederverwendung?

Insgesamt betrachteten wir auch, wie gut die Studierenden Domain Engineering praktizieren und dabei eine umfassende Produktlinie entwickeln

können; allerdings spielte dieser eher edukative Aspekt eine nachgelagerte Rolle für die betrachtete Untersuchung, wenngleich wir auch in diesem Bericht an einigen Stellen darauf eingehen werden.

2.3.2 Szenario

Das Entwicklungsszenario, welches hinter diesen Fragen liegt, ist denkbar einfach: Es gibt zwei Gruppen: Eine repräsentiert den Auftraggeber, die andere den Zulieferer und beide kennen die Kernfunktionalität eines Systems. Beide entwickeln dieses System aber unabhängig voneinander. Das heißt, sie erkunden mögliche Erweiterungen des Systems mit Hilfe von Zukunftsszenarien des Systems und seiner Umgebung gemäß [BF03]. Wichtig dabei ist, dass die Erweiterungen nur in Form von Variationspunkten und Begründungen festgehalten werden. Sie werden nicht vollständig beschrieben oder in den Kerndokumenten implementiert. Der Informationsfluss zwischen diesen beiden Gruppen ist nur sehr gering, da keiner dem anderen zu viel proprietäre Informationen preisgeben möchte. Die eigentliche Schnittstelle ist die Kernspezifikation des Systems und Anmerkungen in der Produktspezifikation, die benennen, welche Teile der Spezifikation instanziierte Variationspunkte sind. Wenn es eine Übereinstimmung der Variationspunkte des Zulieferers und des Auftraggebers gibt, dann werden beide Vorteile hinsichtlich der Wiederverwendung, kürzeren Entwicklungszeiten und einer höheren Qualität davon haben.

Wenn sich außerdem diese lose gekoppelten Produktlinien im Sinne der oben in Frage 2 erläuterten Punkte ähnlich zu einer umfassenden Produktlinie verhält, dann können wir behaupten, dass diese lose gekoppelte Produktlinie auch sinnvoll in Bezug einer Auftraggeber-Zulieferer Beziehung ist und noch weiter geht als andere Wiederverwendungsansätze für Anforderungen, die nur die Qualität und Entwicklungszeit der Auftraggeberspezifikationen adressieren.

Um diese zweite Frage beantworten zu können, haben wir eine dritte Studentengruppe (Vergleichsgruppe) eingeführt, die nur eine einzige, umfassende und vollständige Produktlinie zu entwerfen hatte. Als Basis werden die gleichen Features verwendet, um den Produktlinienkern und die Variationspunkte zu entwickeln. Der Hauptunterschied zu den beiden anderen Gruppen war aber, dass diese Gruppe ihre eigene Produktlinie verwendet, um das endgültige Produkt zu erstellen. Damit gibt es also bezüglich der Produktlinien keinen Bruch mehr zwischen den oberen und unteren Phasen des V-Modells.

2.3.3 Geplanter Ablauf

Wie nun dieses Szenario als kontrolliertes Experiment [JM01, Pre01, Rog95] in ein Praktikum an der Universität Ulm umgesetzt wurde, zeigt dieser Abschnitt. Als erstes konzentrieren wir uns auf die beiden Gruppen, die jeweils die Rollen des Auftraggebers und Zulieferers darstellen; danach wird dann genauer auf die Unterschiede zur „Vergleichsgruppe“ eingegangen.

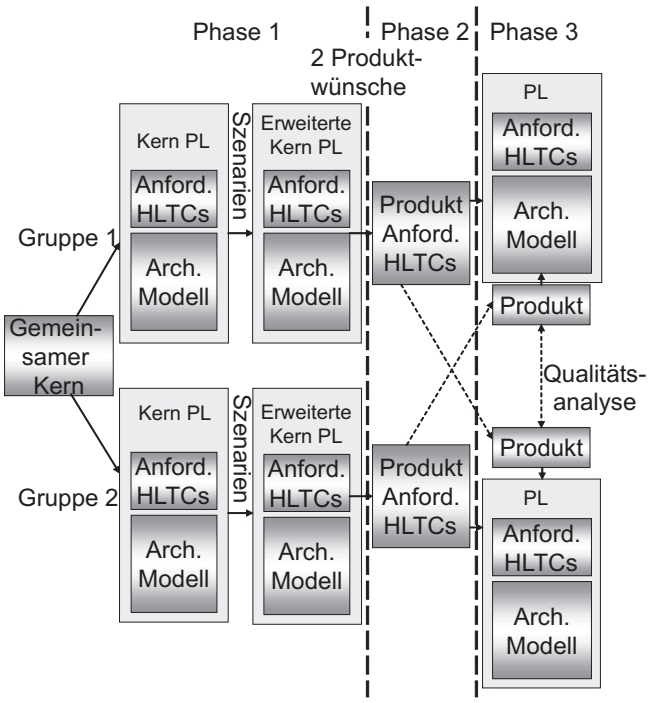


Abbildung 2: Geplanter Ablauf für die Gruppen mit gekoppelten Produktlinien

Abbildung 2 zeigt die geplante Aufteilung der ersten beiden Gruppen, die das Auftraggeber-Auftragnehmer Verhältnis darstellen sollen. Es wäre möglich gewesen, eine Gruppe nur als Auftraggeber und die andere nur als Zulieferer zu deklarieren. Dies hätte aber entscheidende Nachteile

hinsichtlich der Eliminierung möglicher Einflüsse aufgrund des unterschiedlichen Wissenstands und der Motivation der Studierenden gebracht. Daher entschieden wir uns, die Gruppen folgendermaßen zu verdoppeln:

Von einer von uns gestellten, gemeinsamen Kernmenge an möglichen Features ausgehend, beginnt jede Gruppe ihre eigene Produktlinie mit Kern- und Variationspunkten zu entwickeln. Beide Gruppen machen dies sowohl für die unteren als auch für die oberen Phasen des V-Modells, das heißt jede Gruppe ist Auftraggeber und Zulieferer gleichzeitig. Die Anforderungen müssen in Form der FODA-Notation aufgeschrieben werden [KCH+90]. Der dabei entstehende Featurebaum erlaubt es auf einfache Weise, optionale Features — also Variationspunkte — hinzuzufügen (vgl. Abbildung 3). Neben dieser grafischen Notation müssen sie ihre Features und Variationspunkte in einer Tabelle festhalten.

In den „unteren“ Phasen wie Entwurf und Spezifikation wird die Architektur des Produktlinienkerns in Form von Statemate „Activitycharts“ entwickelt. Das Verhalten der Kernfunktionalität wird mit Hilfe von hierarchischen Zustandsautomaten spezifiziert. Darüberhinaus müssen die Variationspunkte in diese Diagramme durch zusätzliche Blöcke (Aktivitäten, Zustände) ohne tatsächlicher Funktionalität integriert werden (vgl. Abbildung 4). Diese Blöcke markieren die Stellen, an denen die Variationspunkte bei Bedarf ausformuliert werden müssen. Sie werden außerdem noch textuell näher erläutert.

Um die Qualität der Modelle zu gewährleisten, werden zwei verschiedene Arten von Testfällen gefordert: Der Auftraggeber erstellt seine eigenen „High-level Testfälle“ (Anwendungstestfälle), der Zulieferer muss Testfälle auf unterer Ebene erstellen (Modultests).

Nachdem diese erste *Domain Engineering* Phase abgeschlossen ist, bekommen die beiden Gruppen zwei unterschiedliche aber gleich umfangreiche Produktwünsche (siehe 2.3.4) von uns gestellt. Während dieser zweiten Phase agiert jede Gruppe in der Rolle des Auftraggebers. Sie müssen den Produktwunsch in ihre Anforderungsdokumente einarbeiten, indem sie die Variationspunkte ausfüllen oder — wo nötig — den Kern ändern. In der dritten Phase dann werden diese „High-level“-Dokumente zwischen den beiden Gruppen ausgetauscht. Ab diesem Zeitpunkt wechseln die Gruppen in die Rolle des Zulieferers und müssen eventuelle Übereinstimmungen zwischen den ausgetauschten Dokumenten, die ja die Auftraggeber-Produktlinie der anderen Gruppe enthalten, und der eigenen Zuliefererproduktlinie in den „Low-level“-Dokumenten finden. Das Abbilden der neuen Anforderungen auf die bereits definierten Variationspunkte sollte viel Aufwand bei der Umsetzung der Produktwünsche vermeiden.

Schließlich werden die fertigen Produkte in Form von Zustandsautomaten wiederum zwischen den beiden Gruppen getauscht und gemäß der vorher definierten Anwendungstestfälle übergeprüft.

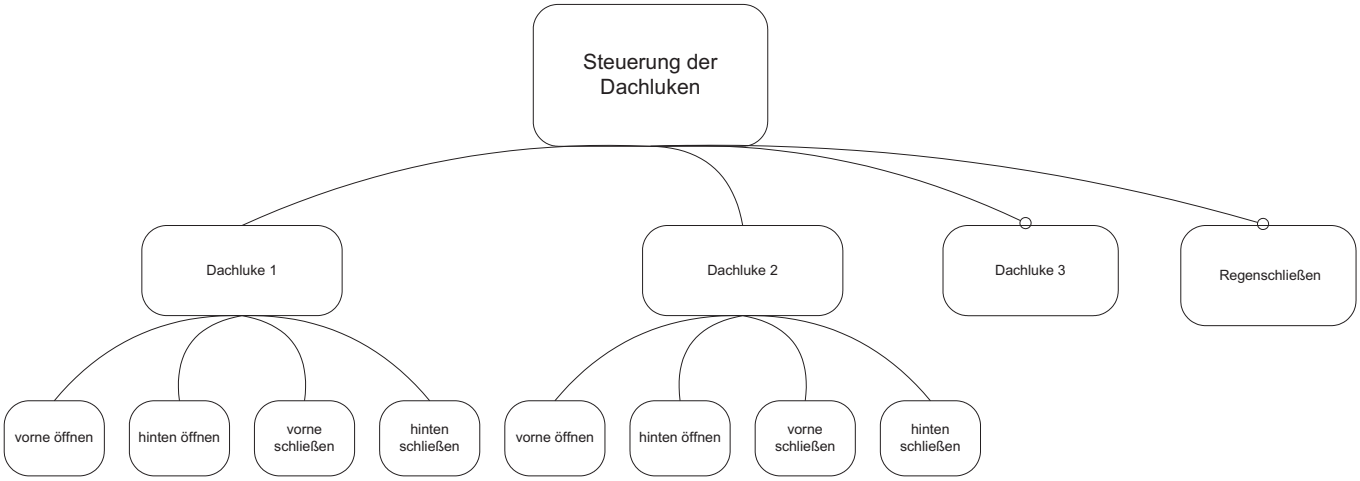


Abbildung 3: Beispiel für eine FODA-Notation

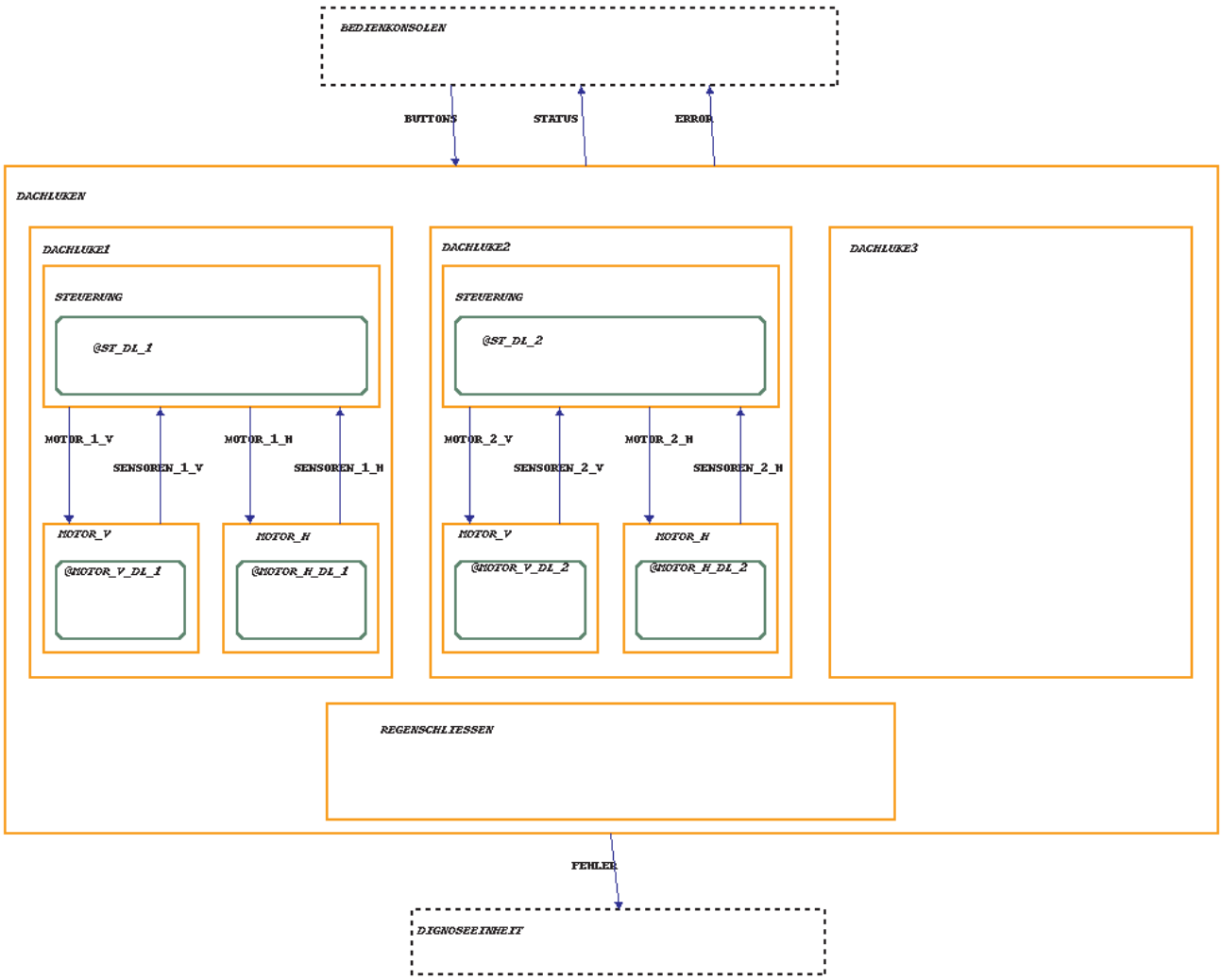


Abbildung 4: Das oben dargestellte FODA-Modell als Statematemodell

Der Ablauf für die Vergleichsgruppe ist im Wesentlichen gleich. Der Hauptunterschied ist, dass sie die Artefakte nicht mit anderen Gruppen tauschen und somit den Produktwunsch in der eigenen Produktlinie zu implementieren haben (in diesem speziellen Fall musste die Vergleichsgruppe beide Produktwünsche umsetzen, da sie aus drei Studierenden bestand, also eine Person mehr umfasste als die beiden anderen Gruppen, siehe Abschnitt 4).

Um für dieses Experiment Aussagen über den Aufwand und das Verhalten der Teilnehmer zu erhalten, wurde eine umfangreiche Aufwandserfassung konzipiert. Jeder Teilnehmer bekommt zu Beginn des Praktikums ein Berichtsheft, das wöchentlich auszufüllen ist und an die Betreuer in elektronischer Form geschickt werden muss (siehe Anhang 2). Die Daten werden während des gesamten Praktikumverlaufs pro Person und Woche gesammelt. Der Aufwand muss nach unseren Vorgaben klassifiziert werden, wobei die Klassifizierung je nach Phase des Experiments unterschiedlich ist. Wir konnten dadurch eine umfangreiche Rohdatensammlung erstellen, um bei der Nachbearbeitung des Praktikums unsere Fragestellungen untersuchen zu können.

Die Aufteilung des Aufwands auf die einzelne Person ist für die Untersuchung der Fragestellungen nicht entscheidend, da wir die Aufwände der Gruppen als ganzes vergleichen wollen. Allerdings bietet sie für die Evaluation des Praktikums wichtiges Wissen, zum Beispiel wie die Aufwandsverteilung über die Zeit pro Person war, die Arbeitsaufteilung innerhalb der Gruppe gestaltet wurde und wie viel Zeitaufwand jeder Teilnehmer in das Prakti-

kum investierte. Die detaillierte Aufwandserfassung dient also nicht nur der Untersuchung des Praktikuminhalts, sondern auch um Informationen über die Durchführung zu erhalten.

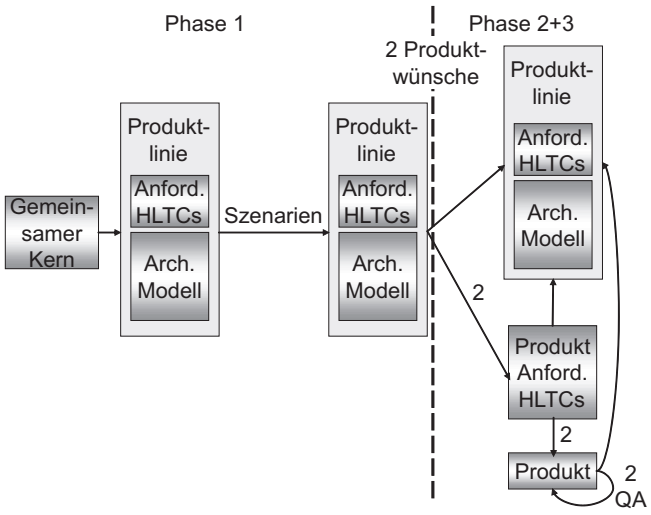


Abbildung 5: Geplanter Ablauf für die Vergleichsgruppe

2.3.4 Beispieldomäne

Wie weiter oben schon erwähnt, haben wir als zu erstellendes Produkt ein Steuergerät für einen Bus ausgewählt. Dieses Steuergerät vereint verschiedenartige Funktionalitäten, die normalerweise über mehrere Steuergeräte verteilt werden würde. Um das Erstellen der Produktlinien aber nicht zu komplex werden zu lassen, wurde alle Funktionalität in nur einem „Bauteil“ integriert. Mit diesem Produkt ließ sich das Verhältnis zwischen Auftraggeber und Zulieferer anschaulich simulieren, so dass der Prozess während des Praktikums von den Studierenden einfach verstanden werden konnte.

Wir haben den Unterschied zwischen Reise- und Linienbus zu Beginn des Semesters bewusst offen gelassen, um die Gruppen nicht in ihrer Kreativität einzuschränken. Für die Kernfunktionalität haben wir eine Spezifikation erstellt (siehe Anhang 3), dessen Funktionalität die Ausgangsbasis für die Produktlinien der Gruppen darstellte. Das Bussteuergerät muss folgende Funktionen in jedem Fall erfüllen:

- Steuerung der Bustüren (vom Fahrer auf- und zuzumachen)
- Steuerung der Klimaanlage (vom Fahrer aus)
- Steuerung des Fahrzeuginnenraumlichts
- Steuerung der Dachluken von der Bedienkonsole beim Fahrer
- Fahrersitzeinstellungen über Knöpfe am Sitz
- Elektrische Einstellung der Außenspiegel über Knöpfe
- Bedienung des Fensters in der Fahrertür
- Speicherung von Fehlern in einer Diagnoseeinheit für Werkstätten

Diese Funktionen eines eingebetteten Systems lassen sich mit Statemate modellieren und simulieren. Hierfür bekommen die Gruppen eine von uns erstellte Simulationsumgebung, mit deren Hilfe sie ihr Modell testen können. Sämtliche Hardwaresignale und Schalter bzw. Anzeigen werden den Gruppen von uns gestellt und sind in der Spezifikation aufgeführt.

Neben der Kernfunktionalität haben wir zwei konkrete Kundenwünsche in Form von Lastenheften konstruiert. Hierzu haben wir zunächst eine Sammlung an möglichen Funktionen erstellt und diesen einen Schwierigkeitsgrad für die Umsetzung zugeordnet. Der Schwierigkeitsgrad wurde aufgrund unserer Erfahrungen bei der Modellierung von eingebetteten Systemen geschätzt. Zugute kam uns hierbei, dass wir ähnliche Aufgaben bereits in vorangegangen Semestern bearbeitet hatten. Auf diese Weise konnten wir zwei unterschiedliche Produktwünsche generieren, die in etwa einen ähnlichen Aufwand in der Umsetzung haben.

Beim Produktwunsch 1 (siehe Anhang 4) handelt es sich um ein Steuergerät für einen Linienbus. Neben der Kernfunktionalität sind folgende Funktionen gewünscht:

- Einführung eines Benutzermanagements für unterschiedliche Fahrer und ihrer jeweiligen Sitzposition
- Steuerung der Bustüren auch von Fahrgästen (unter bestimmten Bedingungen)

- Kopplung des Fahrgastinnenraumlichts an das Abblendlicht des Fahrzeugs
- Integration eines Unfallsensors (bei Stillstand automatische Öffnung der Türen)
- Getrennte Ansteuerung der beiden Flügel der vorderen Doppeltür

Die entsprechenden Erweiterungen für die Hardware sind im Lastenheft für den Produktwunsch 1 aufgeführt.

Für den Produktwunsch 2 (siehe Anhang 5) haben wir das Steuergerät für einen Reisebus ausgelegt. Auch hier gilt: die gesamte Kernfunktionalität bleibt erhalten, allerdings wird die Funktionalität erweitert bzw. an manchen Stellen angepasst. Diese Stellen sind im Einzelnen:

- Einführung eines Benutzermanagements für unterschiedliche Fahrer und ihrer jeweiligen Sitzposition
- Einführung einer Klimaautomatik
- Kopplung der Dachluken an die Scheibenwischer des Fahrzeugs
- Einrichtung einer Außenspiegelheizung
- Einführung eines Komfort-Fensterscheibenhebers

Auch hier werden die erweiterten Hardwaresignale im Lastenheft aufgenommen und den Studierenden übergeben.

Neben der Dokumentation der Aufwände im Berichtsheft sollen die Gruppen ihre Produktlinien und deren Entstehung dokumentieren. Hierzu haben wir zwei Techniken vorgesehen. Zum einen sollen sie eine Feature-Oriented-Domain-Analysis (FODA) für ihre Produktlinie durchführen und die Funktionalität auf diese Weise strukturiert dokumentieren. Zum anderen sollen sie in einer vorbereiteten Excel-Tabelle (Anforderungstabelle) die gefundenen Variationspunkte eintragen und dort mit Begründung klassifizieren.

Um die Dokumentation der gefundenen Variationspunkte vollständig zu machen, müssen diese in die Statematemodelle als Spezifikation aufgenommen werden. Für jeden Variationspunkt soll entsprechend ein neuer Block in das Modell an passender Stelle eingefügt werden. Eine feste Namenskonvention ist hierbei einzuhalten, die Namen für etwaige Signale sollen generisch sein. Im „Beschreibungsfeld“ eines Blocks sollen dann von den Teilnehmern die Daten der Variationspunkte wie beispielsweise die Beschreibung, die ein- und ausgehenden Signale und vor allem die möglichen Abhängigkeiten zwischen den Variationspunkten festgehalten werden.

3 Durchführung des Praktikums

Dieser Abschnitt beschreibt den tatsächlichen Ablauf des Experiments. Zusammen standen uns 13 Wochen für die Untersuchung zur Verfügung — die Zeit, die das Sommersemester 2004 umfasste. In der vorlesungsfreien Zeit haben sich die Betreuer des Praktikums zur Konzeption des Experiments getroffen und die Ziele sowie den geplanten Ablauf geklärt. Das eigentliche Experiment begann dann nach einer Trainingsphase für die sieben Studierenden, in der die wesentlichen Prozesse, Methoden und Tools eingeführt wurden. Die folgenden in Phasen eingeteilten Abschnitte gehen im Detail auf den Ablauf des Experiments ein.

3.1 Einführungsphase - Training

Beim ersten gemeinsamen Treffen führten wir unter den Teilnehmern eine Umfrage mittels eines Fragebogens (siehe Anhang 1) bezüglich deren Vorkenntnisse im Bereich Software Engineering durch. Wie sich herausstellte, hatte keiner der Teilnehmer Vorkenntnisse über das Werkzeug Statemate, sowie über Modellierung im Allgemeinen. Kenntnisse über Zustandsautomaten bzw. hierarchische Zustandsautomaten waren teilweise vorhanden, weil diese in Grundstudiumsveranstaltungen eingeführt wurden. Auf Basis dieser Daten führten wir eine fünfwöchige Schulung mit den Studierenden durch, um sie bis zu Beginn des Experiments auf ein gleiches Wissensniveau zu bringen und ihnen die notwendigen Grundlagen für die erfolgreiche Durchführung des Experiments zu vermitteln. Neben den Grundlagen der Modellierung für die Ver-

wendung von Statemate wurden ihnen auch der verwendete Prozess und die Methoden (wie beispielsweise FODA [KCH+90]) nähergebracht.

Nachdem die Studierenden ein Grundverständnis für die Modellierung mit hierarchischen Zustandsautomaten entwickelt hatten, wurden mit ihnen zwei kleinere Übungsprojekte unter Verwendung von Statemate durchgeführt: zunächst mussten sie eine Zwei-Hand-Presse modellieren und diese mit Hilfe der von uns zur Verfügung gestellten Simulationsumgebung auch testen. Nach diesem sehr einfachen Projekt, wurde es mit der Modellierung und Simulation einer Mikrowelle deutlich schwerer. Neben der Einführung in Statemate wurden aber auch, wie bereits erwähnt, sowohl das Grundlagenwissen über Produktlinien als auch die wichtigsten Aspekte des FODA-Ansatzes anhand eines Beispiels (Aufzugansteuerung) vermittelt. Hierbei wurde den Studierenden auch gezeigt, wie der Produktlinienansatz in Statemate realisiert werden kann (siehe Abschnitt 2.3). Um die Studierenden an die detaillierte Art der Aufwandserfassung zu gewöhnen, mussten sie diese bereits in der Einführungsphase bei der Umsetzung der Beispielprojekte anwenden. Diese Daten wurden von uns erfasst, um einen Überblick über den Zeitaufwand und die Aufwandsverteilung der Studierenden in der Anfangsphase des Praktikums zu erhalten.

Zum Ende der Einführungsphase wurden die Studierenden in drei Gruppen eingeteilt. Für die Gruppenbildung haben wir uns die Vorkenntnisse noch einmal angesehen und auch die gezeigten Ergebnisse in der Einführungsphase mit einfließen lassen, um eine ausgewogene Verteilung der Gruppen für unser Experiment zu erreichen und eine gerechte Kompetenzenverteilung zu garantieren. Uns ist durchaus bewusst, dass dies im Rahmen von kontrollierten Experimenten ungewöhnlich ist, da typischerweise eine zufällige Verteilung der Probanden („Random Sampling“) vorgeschlagen wird. Dadurch hätten wir allerdings auch keine statistische Signifikanz erreicht (da es zu wenig Probanden waren) und uns war es wichtiger, durch eine vergleichbare Verteilung der Kompetenz auch vergleichbare Ergebnisse zu erhalten.

Gruppe 1 und Gruppe 2 bestanden aus je zwei Studierenden, die Vergleichsgruppe bestand aus drei Teilnehmern. Um den Aufwand für alle Beteiligten in etwa ausgewogen zu halten, musste die Vergleichsgruppe entsprechend eine Aufgabe mehr umsetzen: Während Gruppe 1 und 2 je nur einen Kundenwunsch umsetzten, musste die Vergleichsgruppe beide Kundenwünsche realisieren. Dabei lässt sich argumentieren, dass sich durch den Produktlinien-Ansatz die Erstellung neuer Produkte einfacher bewerkstelligen lässt, womit der Aufwand durch die Erstellung des Zweitprodukts nicht verdoppelt, als vielmehr um einen Bruchteil erhöht wird. Nicht zu unterschätzen sind allerdings die durch die dritte Person veränderten Kommunikationsbedürfnisse. Dies führte letztlich dazu, dass wir Frage 2 der oben beschriebenen Motivation im Rahmen dieses Experiments nicht beantworten konnten, da die Aufwände nicht zu vergleichen waren (siehe auch Kapitel 5).

3.2 Das Experiment

Das Experiment im Praktikum verlief — wie bereits geschildert — in drei Experimentphasen und einer Qualitätssicherungsphase ab. Für das Experiment waren insgesamt acht Wochen vorgesehen.

In der ersten Phase bekamen alle drei Gruppen eine Spezifikation für die Kernfunktionalität des Steuergeräts eines Busses (siehe Anhang 3). Die Gruppen hatten dann vier Wochen Zeit, diese Spezifikation in ein Modell in Statemate umzusetzen, also die beschriebene Funktionalität in Statemate zu modellieren und zu testen. Neben dieser Aufgabe mussten sich die Gruppen Gedanken über die Ausweitung der Funktionalität hin zu einer Produktlinie machen. Die Anforderungen an diese Produktlinien wurden mittels FODA analysiert und dokumentiert. Die Teilnehmer machten sich hierfür Gedanken über zukünftige Szenarien und hielten Erweiterung in Form von Variationspunkten fest. Zusätzlich wurden die geplanten Funktionalitäten als Variationspunkte in die Excel-Anforderungstabelle aufgenommen und dort näher als zusätzliche Funktionalität oder als Alternative spezifiziert. Die gefundenen Szenarien dienten hierbei als Begründung für die jeweiligen Variationspunkte. Am Ende der ersten Phase (nach vier Wochen) hatten die Gruppen jeweils die Kernfunktionalität in Statemate umgesetzt und Produktlinien für das Produkt „Steuergerät eines Busses“ er-

stellt. Die Variationspunkte waren schriftlich dokumentiert und in die Statecharts eingearbeitet. Um sicherzustellen, dass die Gruppen die Aufgabe verstanden hatten, wurden in den vier Wochen regelmäßige Treffen abgehalten und informelle Reviews seitens der Betreuer durchgeführt und mit der jeweiligen Gruppe über deren Ergebnisse diskutiert. Auf diese Weise konnten wir den Fortschritt bei der Erstellung der Produktlinien — insbesondere in Hinblick darauf, an welche erweiterten Funktionalitäten gedacht wurde — beobachten.

In der fünften Woche begann die zweite Phase des Experiments. Ab diesem Zeitpunkt unterschied sich die Aufgabenstellung für die Vergleichsgruppe von denen der anderen beiden Gruppen.

Die Vergleichsgruppe hatte bis zum Ende des Praktikums Zeit, beide Produktwünsche mit einer durchgängigen Produktlinie umzusetzen. Sie musste die Produktwünsche analysieren, in ihre Produktlinie einbauen und entsprechend Variationspunkte ausmodellieren bzw. unter Umständen die Kernfunktionalität anpassen (vgl. Abschnitt 2.3).

Für die Gruppen 1 und 2 ging das Praktikum folgendermaßen weiter: jede Gruppe bekam jeweils einen Produktwunsch und hatte dann eine Woche Zeit, diesen Wunsch auf Anforderungsebene in ihre Produktlinie einzuarbeiten, was bedeutete, dass sie den Wunsch analysieren, ihre Feature- und Kontextdiagramme anpassen, die betroffenen Variationspunkte bestimmen und in der Anforderungstabelle aktualisieren mussten. Dies führte bei beiden Gruppen zu einer Spezifikation der Produktwünsche auf Seiten des Automobilherstellers, also in der oberen Hälfte des V-Modells. Zusätzlich mussten High-Level-Testfälle auch für die erweiterte bzw. geänderte Funktionalität erstellt werden, mit deren Hilfe das spätere Produkt auf das richtige Verhalten getestet werden sollte. Hierzu hätten die von Statemate angebotenen Simulation Control Programs (SCPs) genutzt werden sollen, die es ermöglichen, Testabläufe in einer spezifischen Sprache zu notieren und dann immer wieder automatisiert ablaufen zu lassen. Allerdings zeigte sich, dass der Aufwand für das Erstellen dieser Skripte zu hoch für die Studierenden war. Deshalb wurde auf eine textuelle Darstellung ausgewichen.

Nach dieser Woche wurden die Dokumente und die Produktwünsche zwischen den beiden Gruppen getauscht. Die Gruppen mussten nun aufgrund der Spezifikation des ihnen fremden Produktwunsches der jeweils anderen Gruppe diese in einem Produkt umsetzen, also in ihr Statematemodell einzuarbeiten. Dabei ist zu beachten, dass die Spezifikation auf Grundlage der von der anderen Gruppe erstellten Produktlinie basiert; damit ging aus der Spezifikation hervor, welche Teile umgesetzte Variationspunkte waren und welche Teile ursprünglich zum Kern gehörig angesehen waren. Dies unterstützte die Produkt-erstellende Gruppe, da sie das Wissen über die Variationspunkte teilweise auf ihre Produktlinie übertragen konnte. Für diese Aufgabe hatten beide Gruppen jeweils zwei Wochen Zeit. In dieser Phase musste von beiden Gruppen mit Hilfe von Software-Testfällen auf Zuliefererseite sichergestellt werden, dass ihr Produkt den jeweiligen Anforderungen entsprach.

In der letzten Woche des Experiments (Qualitätssicherungsphase) tauschten die Gruppen ihre Produkte wieder zurück und übernahmen nun als Auftraggeber die Rolle der Qualitätsprüfung, das heißt, sie testeten das ihnen gelieferte Produkt mit ihren abstrakteren Testfällen und führten eine Abnahmeprüfung durch. Die Ergebnisse wurden in Testprotokollen dokumentiert.

Die Vergleichsgruppe musste bis eine Woche vor Ende des Praktikums beide Kundenwünsche mit ihrer Produktlinie realisiert haben und in der letzten Woche ebenfalls einen Abnahmetest durchführen.

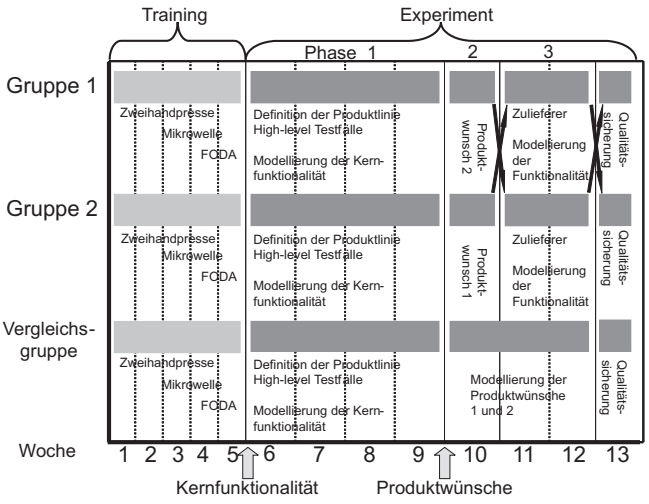


Abbildung 6: Schematischer Ablauf des Praktikums

Zur besseren Übersicht wird in Abbildung 6 der gesamte Ablauf des Praktikums schematisch dargestellt. In der x-Achse ist der Verlauf nach Wochen eingeteilt.

Zu sehen ist die fünfwöchige Einführungsphase und das acht Wochen dauernde Experiment. Durch die Pfeile zwischen Gruppe 1 und Gruppe 2 wird der Tausch der Anforderungen in Woche 10 und der Rücktausch der fertigen Produkte in der Woche 12 dargestellt.

3.3 Nachbereitung

Nach dem Ende des Praktikums wurden die während der Durchführung beobachteten Daten gesammelt und analysiert. Für die Analyse der Durchführung des Experiments standen folgende Rohdaten zur Verfügung:

- Die detaillierten Aufwandserfassungbögen eines jeden Teilnehmers, aufgelistet in den Praktikumsberichten (siehe beispielhaft Anhang 2)
- Die Zukunftsszenarien jeder Gruppe (mit den spezifischen Anforderungen in tabellarischer Form)
- Die Anforderungsdokumentation in FODA-Notation
- Die Statematemodelle jeder Gruppe (mit dem Design der Variationspunkte und der Umsetzung der jeweiligen Produktwünsche)
- Die erstellten Testfälle
- Die in den Testprotokollen aufgeführten Testergebnisse (Qualitätssicherung)

Auf der Basis dieser Rohdaten wurden die aus dem Experiment erhaltenen Ergebnisse analysiert und in Beziehung zu den eingangs erwähnten Zielen des Experiments gesetzt. Die Resultate werden im folgenden Abschnitt diskutiert.

4 Ergebnisse

Um die Frage untersuchen zu können, ob Variationspunkte auf Ebene der Anforderungen auf Variationspunkte auf der Code- und Architekturebene abgebildet werden können, haben wir uns die gefundenen Variationspunkte der einzelnen Gruppen angesehen und diese in Beziehung zueinander gesetzt.

Zu Beginn der Vorbereitungen für das Praktikum, als wir uns entschieden haben, ein Bussteuergerät als Untersuchungsobjekt zu verwenden, haben wir selbst nach möglichen Szenarien und Erweiterungen gesucht. Wir haben 13 mögliche und sinnvolle Variationspunkte (oder in anderen Worten, 13 mögliche Anforderungen) gefunden, welche die Kernfunktionalität erweitern können. Natürlich sind diese von uns gefundenen Punkte nicht vollständig.

Wir verglichen nun unsere Variationspunkte mit denen der verschiedenen Gruppen. Tabelle 1 zeigt die von den Gruppen gefundenen Variationspunkte: In der zweiten Spalte *Betreuer VPs* wird dargestellt, wie viele der 13 von den Betreuern gefundenen Variationspunkte von der jeweiligen Gruppe gefunden wurden. In der dritten Spalte *Eigene VPs* ist die Anzahl der Variationspunkte aufgelistet, die die Gruppen zusätzlich entdeckt haben. V-Gruppe steht hierbei für die Vergleichsgruppe.

Gruppe	Betreuer VPs	Eigene VPs	$\sum$
Gruppe 1	9	4	13
Gruppe 2	9	4	13
V-Gruppe	7	4	11

Tabelle 1: Anzahl der gefundenen Variationspunkte.

Gruppe 1 hat insgesamt 13 mögliche Variationspunkte entdeckt, davon vier unterschiedlich zu unseren. Auch die zweite Gruppe hat 13 Variationspunkte entdeckt, wiederum waren davon vier unterschiedlich zu unseren und auch zu denen der Gruppe 1. Die Vergleichsgruppe kam auf elf Variationspunkte, vier davon unterschiedlich zu den von uns gefundenen. Drei Variationspunkte dieser Gruppe wurden von keiner anderen Gruppe gefunden. Alles in allem wurden 24 Variationspunkte gefunden, wobei ein jeder von mindestens einer Gruppe gefunden wurde.

In dieser Arbeit beschränken wir uns auf die 13 Variationspunkte, welche bei der Vorbereitung des Praktikums von den Betreuern gefunden wurden, da diese die Basis für das Experiment darstellen. Nur einer dieser 13 Variationspunkte wurde von keiner Gruppe gefunden, alle anderen wurden zumindest von einer Gruppe entdeckt: Vier der Variationspunkte wurden von jeder Gruppe entdeckt, fünf von jeweils zwei Gruppen und schließlich wurden drei Variationspunkte von jeweils nur einer Gruppe identifiziert.

Nach der zehnten Woche im Praktikum realisierten die Gruppen die um Erweiterungen angereicherten Produktwünsche. Sehr interessant für unsere Ergebnisse sind die gefundenen Variationspunkte der Gruppen im Zusammenhang mit den Produktwünschen: wie viele Anforderungen aus den Produktwünschen können von den Gruppen auf ihre Variationspunkte abgebildet werden, müssen die Gruppen ihre Kernfunktionalität ändern und wie wirkt sich das auf den zeitlichen Aufwand aus?

In der Phase 2 des Experiments bekam jede der Gruppen 1 und 2 einen anderen Produktwunsch und musste daraus eine Spezifikation erstellen. In jedem dieser Produktwünsche mussten jeweils fünf von den 13 Anforderungen der Betreuer umgesetzt werden. Hierbei unterschieden sich die Anforderungen der beiden Produktwünsche komplett, so dass insgesamt zehn Variationspunkte umzusetzen waren. Gruppe 1 hatte alle fünf für ihren Produktwunsch relevanten Anforderungen in Variationspunkten identifiziert, so dass es für sie relativ einfach war, das konkrete Produkt aus ihrer Produktlinie heraus zu spezifizieren, indem sie die nötigen Variationspunkte ausgestaltete. Die Gruppe 2 konnte nur vier der Anforderungen auf ihre Variationspunkte abbilden, die Mitglieder mussten folglich die Kernfunktionalität ändern, um die Produktspezifikation zu erstellen. Tabelle 2 fasst diese Werte in Spalte zwei zusammen.

Gruppe	Phase 2	Phase 3
Gruppe 1	5/5 (100%)	2/5 (40%)
Gruppe 2	4/5 (80%)	4/5 (80%)
V-Gruppe	—	3/5 (60%)
		und 2/5 (40%)

Tabelle 2: Anzahl der abgedeckten Variationspunkte pro Produktwunsch und Gruppe.

In der dritten Phase unseres Experiments wurden die Anforderungsspezifikationen von beiden Gruppen ausgetauscht (siehe Abschnitt 2.3). Nun mussten die Gruppen die Spezifikation der jeweils anderen Gruppe umsetzen. Gruppe 1 identifizierte zwei der fünf Variationspunkte aus der Spezifikation von Gruppe 2, so dass sie lediglich zwei ihrer Variationspunkte benutzen konnten, um das Produkt zu implementieren. Um das Produkt aus ihrer Produktlinie gestalten zu können, mussten sie die Kernfunktionalität für drei Anforderungen anpassen. Die Mitglieder von Gruppe 2 identifizierten vier von fünf Variationspunkten aus dem Produktwunsch. Aus diesem Grund war es für sie relativ einfach möglich, das Produkt zu instanziieren, indem sie die vier Variationspunkte ausarbeiteten.

Die Aufgabe für die Vergleichsgruppe unterschied sich von denen der beiden anderen Gruppen, wie bereits in Abschnitt 2.3 erwähnt. Die Mitglieder der Gruppe mussten beide Produkte aus ihrer Produktlinie heraus realisieren. Sie identifizierten drei Variationspunkte vom ersten Produktwunsch und und zwei der fünf vom zweiten Produktwunsch. Die Resultate werden in Tabelle 2 nochmals zusammengefasst. Die Vergleichsgruppe musste lediglich die Produktwünsche umsetzen, deswegen haben wir auch nur die Ergebnisse aus Phase 3, aber dafür von beiden Produktwünschen. Die Notation *x/y* bedeutet: die Gruppen konnten x von y Variationspunkten abdecken. Der Wert in Klammern drückt die Abdeckung in Prozent aus.

Die Anforderungen, und damit einhergehend die Variationspunkte, waren von unterschiedlicher Komplexität. Jeder Produktwunsch enthielt zwei Anforderungen hoher, eine Anforderung mit mittlerer und zwei Anforderungen mit geringer Komplexität bezüglich der Umsetzung. Diese Tatsache beeinflusst natürlich die Ergebnisse unserer Untersuchung, da es wichtig ist, ob die Gruppen Variationspunkte mit hoher oder niedriger Komplexität abdecken konnten. Um dies zu illustrieren, gaben wir jedem der zehn Variationspunkte einen zusätzlichen Gewichtungsfaktor: +5 für hohe Komplexität, +3 für mittlere und +1 für geringe Komplexität. Jeder Produktwunsch hat demzufolge einen Gewichtungsfaktor von 15 ($2 \cdot 5 + 1 \cdot 3 + 2 \cdot 1$). Tabelle 3 listet die abgedeckten gewichteten Variationspunkte jeder Gruppe auf.

Gruppe	Phase 2	Phase 3
Gruppe 1	15/15 (100%)	10/15 (67%)
Gruppe 2	14/15 (93%)	14/15 (93%)
V-Gruppe	—	13/15 (87%)
		und 8/15 (53%)

Tabelle 3: Abgedeckte gewichtete Variationspunkte pro Gruppe und Produktwunsch.

Interessant für unsere Fragestellung ist hauptsächlich die Phase 3, die Implementierungsphase, in welcher das Produkt aus der Produktlinie heraus realisiert wird. Spalte drei von Tabelle 3 zeigt die Schlüsseldaten: Gruppe 1 konnte ungefähr 67% der Anforderungen auf ihre Variationspunkte abbilden, wenn man den Gesamtaufwand mit den gewichteten Anforderungen berücksichtigt. Die Gruppe 2 konnte 93% der Anforderungen aus ihrem Produktwunsch mit Variationspunkten abdecken. Die Vergleichsgruppe, welche ja beide Kundenwünsche realisieren musste, konnte 87% der Anforderungen des ersten Kundenwunsches und 53% der Anforderungen des zweiten Kundenwunsches mit ihren Variationspunkten abdecken. Fokussierend auf die Frage 1 unserer Untersuchung (siehe Abschnitt 2.3), war es der Gruppe 1 möglich, 67% der Anforderungen, die in der Produktspezifikation von Gruppe 2 definiert wurden, mit ihren Variationspunkten abzudecken. Gruppe 2 schaffte es sogar, 93% der aus der Produktspezifikation von Gruppe 1 kommenden Anforderungen mit ihren Variationspunkten abzudecken.

Es ist notwendig, diese Ergebnisse nun mit den Daten, die aus der detaillierten Aufwandserfassung gewonnen wurden, zu vergleichen. Wir nahmen die Rohdaten, die von jedem Teilnehmer der Untersuchung vorlagen, und führten eine Analyse durch. Dies erlaubte uns, Daten über die Gruppenarbeit und ihren Aufwand bezüglich der unterschiedlichen Aufwandskategorien zu sammeln. Interessant für diese Arbeit ist der Aufwand, welcher die Umsetzung der Produkte aus der Produktlinie betrifft, unterteilt nach verschiedenen Kategorien. Wir definierten drei Aufwandskategorien für diese Phase:

- A: Aufwand für die Instanziierung eines Variationspunktes
- B: Aufwand für das Ändern bzw. Neumodellieren der Kernfunktionalität
- C: Aufwand für zusätzlichen Modellierungsaufwand ohne Bezug zur Kategorie A oder B

Die Teilnehmer notierten ihren Aufwand in Minuten, wobei sie ihren Aufwand in die Kategorien A, B oder C kategorisierten (und natürlich auch in andere Kategorien, die aber an dieser Stelle keinen Einfluss haben — beispielsweise Aufwand für Verständnis). Tabelle 4 beinhaltet die entsprechenden Ergebnisse der Aufwandsanalyse: den Aufwand für die Umsetzung der Produkte eingeteilt in die erwähnten Kategorien, ausgedrückt in Prozent.

Gruppe	Kat. A	**Kat. B**	Kat. C
Gruppe 1	64.2%	**16.8%**	19%
Gruppe 2	69.6%	**8.7%**	21.7%
V-Gruppe	8.2%	**57%**	24.8%

Tabelle 4: Kategorisierter Aufwand in Prozent in Phase 3 pro Gruppe.

Die Mitglieder von Gruppe 1 verbrachten 64.2% ihres Aufwands bei der Erstellung des Produkts mit dem Ausarbeiten der Variationspunkte, 16.8% mit der Anpassung ihrer Kernfunktionalität und 19% der Zeit mit dem Modellieren ohne Bezug zu den Kategorien A oder B. Diese Werte passen zu den Daten, die aus der Anzahl der entdeckten gewichteten Variationspunkte gewonnen wurden (siehe Tabelle 3). Gruppe 2 benötigte nur 8.7% der gesamt eingesetzten Zeit um die Kernfunktionalität anzupassen, 69.6% des Aufwands wurden in die Ausarbeitung der Variationspunkte gesteckt. 21.7% ihres Aufwands benötigten die Mitglieder dieser Gruppe für die Implementierung ihres Produkts, ohne ihn der Kategorie A oder B zuordnen zu können.

Die Resultate der Vergleichsgruppe überraschten uns: obwohl die Mitglieder dieser Gruppe es schafften, 87% der Anforderungen des ersten Produktwunsches und 53% des zweiten Produktwunsches auf Variationspunkte ihrer Produktlinie abzubilden, benötigten sie die meiste Zeit während der Implementierung — mit 57% über die Hälfte — für die Anpassung ihrer Kernfunktionalität. Lediglich 8.2% ihrer verbrauchten Zeit wurde für die Ausarbeitung von Variationspunkten genutzt, die restliche Aufwand war ohne Bezug zu Kategorie A oder B. Wie konnte das passieren? Wir fanden bei der Analyse ihrer Produktliniendokumente heraus, dass diese Gruppe die Kernfunktionalität falsch modelliert hatte. Sie unterschieden zwar die Kernfunktionalität und die Variationspunkte kor-

rekt, jedoch implementierten sie die Kernfunktionalität inkorrekt. Diese Tatsache bemerkten die Teammitglieder erst in Phase 3 unserer Untersuchung und mussten deshalb in dieser Phase ihre Kernfunktionalität remodellieren und den Kern ihrer Produktlinie neu implementieren. Dieser Fehler macht es leider unmöglich, die zweite Fragestellung (siehe Abschnitt 2.3) unseres Experimentes zu beantworten.

Um die Frage zu beantworten, warum die Mitglieder der Vergleichsgruppe ihren Fehler nicht früher bemerkten, verglichen wir die Zeiten, die jede Gruppe mit der Qualitätssicherung ihrer Produktlinie während des Experiments verbracht hatte. Gruppe 1 wandte 21% ihres Aufwands für die Qualitätssicherung auf, Gruppe 2 steckte 10% des Aufwands in die Qualitätssicherung betreffende Aufgaben. Die Mitglieder der Vergleichsgruppe jedoch investierten in die Qualitätsüberprüfung ihrer Modelle einen verschwindend kleinen Anteil ihres Aufwands von 2%: sie versäumten es, sicherzustellen, dass ihre Kernfunktionalität korrekt implementiert wurde, was zur Notwendigkeit des Remodellierens führte.

Zum Abschluss unserer Analyse schauten wir Betreuer uns die Modelle und Testberichte, die in der letzten Phase der Untersuchung erstellt wurden, an. Die Qualität der konkreten Produkte, die aus den jeweiligen Produktlinien erstellt wurden, war gut: die Produkte erfüllten alle Testanforderungen. Kleinere Mängel konnten mit wenig Aufwand sehr rasch beseitigt werden, so dass als Resumé gezogen werden kann, dass die von den Zulieferern an den Auftraggeber gelieferten Produkte eine gute Qualität hatten.

Die kostenintensiven Arbeiten in Bezug auf Geld und Zeit in einem Produktlinienumfeld sind Änderungen an der Kernfunktionalität. Je mehr Variationspunkte von Anforderungen eines Produkts betroffen sind, desto leichter kann das Produkt aus der Produktlinie heraus umgesetzt werden. Wenn der Zulieferer eine substanzielle und ausgeklügelte Produktlinie hat und die Anforderungen von Seiten des Auftraggeber auf Variationspunkte abbilden kann, ist die Instanziierung des Produkts sehr ökonomisch bezüglich Zeit und Geld möglich. Das liegt wie bereits erwähnt daran, dass die Kernfunktionalität der Produktlinie nicht betroffen ist. Dies kann man auch erkennen, wenn man die Daten, gewonnen in Phase 2, aus Spalte zwei der Tabelle 3 und aus Spalte zwei der Tabelle 4 (fett markiert) für die Gruppen 1 und 2 miteinander vergleicht. Gruppe 1 schaffte es, 67% der Anforderungen auf Variationspunkte abzubilden und benötigte 16.8% des Aufwands, um entsprechend die Kernfunktionalität noch anzupassen. Den Mitgliedern der Gruppe 2 gelang es, sogar 93% der Anforderungen auf ihre Variationspunkte abzubilden, deswegen mussten sie nur 8.7% ihrer Implementierungszeit für die Änderungen an der Kernfunktionalität verwenden.

Ein interessanter Aspekt in diesem Umfeld ist sicherlich die Abbildung der Produktanforderungen auf Variationspunkte. In unserem Experiment wurden dem Zulieferer die Anforderungen in Form einer Produktspezifikation, abgeleitet aus einer Produktlinie, zur Verfügung gestellt. Mit unserem Experiment können wir leider keine Aussage machen, ob die Kommunikation zwischen Auftraggeber und Zulieferer bei diesem Ansatz einen Vorteil hat, da wir keine Vergleichsdaten bezüglich einer anderen Kommunikationsform haben. Wir denken, dass eine Untersuchung, welche die produktlinienunterstützte Kommunikation mit anderen Kommunikationsarten vergleicht, sehr schwierig und im universitären Umfeld gar unmöglich wäre. Ein entsprechend positiver Effekt würde wohl nur bei einer hohen Anzahl von Anforderungen und damit einhergehender Größe der Anforderungsdokumente auftreten, was in einem zeitlich sehr beschränkten Praktikum aber nicht durchführbar ist. Die aktuelle Situation in der Praxis ist der Austausch von Anforderungen mittels herkömmlichen Textdokumenten. Wir halten es für nötig, dass diese Form der Kommunikation verbessert werden sollte, und wir können darauf hinweisen, dass der Austausch von Anforderungen zwischen dem Auftraggeber und dem Zulieferer mit dem hier vorgestellten Ansatz der gekoppelten Produktlinie möglich ist. Der Mehrwert des Einsatzes einer Produktlinie auf Ebene des Auftraggebers, um sie unter anderem auch für die Kommunikation mit dem Zulieferer zu nutzen, ist ein möglicher Gegenstand weiterer Forschungstätigkeit.

5 Kritische Diskussion des Experiments und Ausblick

Das Experiment gibt Hinweise, dass die beschriebenen, lose gekoppelten Produktlinien grundsätzlich möglich sind. Dennoch ließ sich eine Frage im Verlauf dieses Experiments nicht klären: Zu welchem Ausmaß verhält sich eine gekoppelte Produktlinie wie eine vollständige (Frage 2 der motivierenden Fragestellungen zu diesem Experiment). Dies wird zum Teil deswegen begründet, weil die Vergleichsgruppe, obwohl sie einen ähnlichen Wissensstand des Software Engineering wie die beiden anderen Gruppen hatte, diesen nicht sinnvoll nutzen konnte, weil sie sich effektiv in die drei Studierenden aufgeteilt hatte, so dass Gruppensynergien nicht auftraten. Da jeder Student eine eigene Aufgabe bearbeitete und sie sich nur an den Schnittstellen untereinander austauschten, generierten sie kein gemeinsames Bild der Produktlinie und vor allem deren Kerns, so dass sie bei der Produkterstellung erheblich mehr Aufwand hatten, weil sie eben nicht auf eine stabile Basis zurückgreifen konnten. An dieser Stelle hätte man zweierlei zur besseren Durchführung des Experiments beitragen können: Zunächst hätten wir als Betreuer verstärkt Einfluss auf die Gruppe ausüben sollen, die Arbeiten regelmäßig gemeinsam zu erledigen. Trotzdem wir aus den uns vorliegenden Daten zur Aufwandserfassung einen erheblichen Unterschied zu den anderen Gruppen feststellten, so war dies während des Experiments noch unklar, ob dies am Mehraufwand und der weiteren Person in der Gruppe oder einem falschen Vorgehen lag. Des weiteren hätte man auch durch eine Änderung im Experimentdesign hier eingreifen können: Da in diesem Fall eines kontrollierten Experiments die in Betracht stehenden Methoden sich kaum unterschieden (letztlich war nur das Szenario, in denen sie eingebettet waren, unterschiedlich), wie es normalerweise häufig der Fall ist, hätte man durch einen Tausch der Rollen der Gruppen eine Aussage zu dieser Fragestellung machen können. Hätte man das Experiment mit denselben Probanden und einem anderen, aber vergleichbaren System noch einmal durchgeführt, wobei die Tausch-Gruppen die Rolle der Vergleichsgruppe eingenommen hätten und umgekehrt, so wäre ein besserer Vergleich zwischen dem gekoppelten Produktlinien-Ansatz sowie einem vollständigen Produktlinien-Ansatz möglich gewesen.

Eine weitere Limitation des Experiments liegt darin, dass wir keinen Vergleich durchführen konnten zwischen den Tausch-Gruppen, die ja einen Produktlinien-Ansatz lebten, mit Tausch-Gruppen, die nicht in einem Produktlinien-Verfahren entwickelt hatten. Dies wäre notwendig gewesen, um die Stärke des Produktlinien-Ansatzes in diesem Kontext selbst zeigen zu können. Allerdings war dies im Rahmen dieses universitären Praktikums nicht möglich gewesen. Produktlinien-Ansätze machen erst ab einer gewissen Größe der Systeme und dem Zwang zur Wiederverwendung Sinn. Dies konnten wir nicht simulieren; die im Praktikum entwickelten Systeme waren klein und Wiederverwendung war keine zwingende Voraussetzung, um diese sinnvoll implementieren zu können. Daher mussten wir von allen Beteiligten ein Produktlinien-Vorgehen verlangen, da andernfalls die Gruppen, die ohne Produktlinien vorgegangen wären, sehr viel weniger Aufwand gehabt hätten. Betrachtet man allerdings die heutige Praxis bei der Erstellung der Spezifikationen und die Übergabe dieser an Zulieferer, so stellt man fest, dass es gerade an dieser Schnittstelle Probleme gibt, vor allem mit solcherart Anforderungen, die große Teile schon vorhandenen Software-Designs der Zulieferer zerstören würden. Würde konsequent nach dem hier beschriebenen Vorgehen entwickelt werden, so wären diese Probleme an der Schnittstelle zwischen Auftraggeber und Zulieferer Vergangenheit.

Weitere Grenzen des Experiments, die wir oben schon erwähnt haben, lassen sich auch auf den universitären Rahmen zurückführen, in dem es durchgeführt wurde: Zunächst einmal war die Anzahl an Teilnehmern begrenzt, so dass statistisch signifikante Resultate nicht abzuleiten sind. Weiterhin konnte nicht zu 100% ausgeschlossen werden, dass die Studierenden sich über ihre Inhalte austauschten, als sie in der Phase waren, ihre Szenarien für die Weiterentwicklung des Systems zu generieren. Nicht zuletzt spielte auch unsere Rolle als Betreuer in das Experiment, da wir die essentiellen, zu erstellenden Produktwünsche und damit Variationspunkte vorgaben. Dies war uns bewusst, dennoch erschien uns dieses Vorgehen besser als die Alternative, die Studierenden selber die Produktwünsche generieren zu lassen, da in die-

sem Fall äußerst unwahrscheinliche Weiterentwicklungen denkbar gewesen wären, die wir ausschließen wollten. Um dennoch einem aktiven Eingreifen in den Experimentverlauf vorzubeugen, haben wir die von uns ausgegebenen Dokumente, inklusive der Produktwünsche, in der Zeit vor dem Experimentbeginn erstellt.

Im Experiment konnten nur die ersten Instanzen bei der Arbeit mit Produktlinien aufgezeigt werden. Wir gehen davon aus, dass mit intensiverer Verwendung von Produktlinien und mit wachsender Erfahrung der Entwickler, der Kern und die Variationspunkte sehr viel stabiler werden.

Trotz der aufgezeigten Grenzen des Experiments sind die Ergebnisse viel versprechend, da bisher in diesem Kontext keine empirische Forschung durchgeführt wurde. Was man aus dem Experiment lernen kann, ist, dass die Forschung sich mit der Thematik gekoppelter Produktlinien befassen sollte, um einen Gewinn sowohl für die Auftraggeber als auch die Zulieferer zu bieten.

Literatur

[ABM00] Atkinson, C., J. Bayer und D. Muthig: *Component-Based Product Line Development: The KobrA Approach.* In: Donhoe, P. (Herausgeber): *Software Product Lines: Experiences and Research Directions*, Seiten 289–309, Norwell, 2000.

[AG01] Anastasopoulos, M. und C. Gacek: *Implementing Product Line Variabilities.* In: *Proceedings of the Seventh Symposium on Software Reusability: Putting Software Reuse in Context*, Seiten 109–117, Toronto, 2001.

[AHKSC01] Auerswald, M., M. Herrmann, S. Kowalewski und V. Schulte-Coerne: *Reliability-Oriented Product Line Engineering of Embedded Systems.* In: Linden, F. van der (Herausgeber): *Proceedings of the 4^{th} International Workshop on Software Product-Family Engineering (PFE)*, Seiten 83–100, Bilbao, 2001.

[AM02] Atkinson, C. und D. Muthig: *Enhancing Component Reusability through Product Line Technology.* In: Gacek, C. (Herausgeber): *Proceedings of the 7^{th} International Conference on Software Reuse: Methods, Techniques, and Tools*, Seiten 94–108, Austin, 2002.

[BCK03] Bass, L., P. Clements und R. Kazman: *Software Architecture in Practice.* The SEI Series in Software Engineering, Addison-Wesley, Boston, 2003.

[BF03] Bush, D. und A. Finkelstein: *Requirements Stability Assessment Using Scenarios.* In: *Proceedings of the 11^{th} IEEE International Conference on Requirements Engineering (RE)*, Seiten 23–32, 2003.

[BFK+99] Bayer, J., O. Flege, P. Knauber, R. Laqua, D. Muthig, K. Schmid, T. Widen und J.-M. DeBaud: *PuLSE: A Methodology to Develop Software Product Lines.* In: *Proceedings of the Fifth Symposium on Software Reusability: Bridging the Gap between Research and Practice*, Seiten 122–131, Los Angeles, 1999.

[BHJ+03] Birk, A., G. Heller, I. John, T. von der Massen, K. Müller und K. Schmid: *Product Line Engineering Industrial Nuts and Bolts.* Technischer Bericht 113.03/E, Fraunhofer IESE, Kaiserslautern, 2003.

[Böc00] Böckle, G.: *Model-Based Requirements Engineering for Product Lines.* In: Donhoe, P. (Herausgeber): *Software Product Lines: Experiences and Research Directions*, Seiten 193–203, Norwell, 2000.

[Bos00a] Bosch, J.: *Design and Use of Software Architectures: Adopting and evolving a product-line approach.* Addison-Wesley, ACM Press Books, Harlow, 2000.

[Bos00b] Bosch, J.: *Organizing for Software Product Lines.* In: Linden, F. van der (Herausgeber): *Proceedings of the International Workshop on Software Architectures for Product Families*, Seiten 117–134, LasPalmas de Gran Canaria, 2000.

[CDKT01] Chastek, G., P. Donhoe, K.C. Kang und S. Thiel: *Product Line Analysis: A Practical Introduction.* Technischer Bericht CMU/SEI-2001-TR-001, Software Engineering Institute of the Carnegie Mellon University, Pittsburgh, 2001.

[CN02] Clements, P. und L. Northrop: *Software Product Lines: Practices and Patterns.* The SEI Series in Software Engineering, Addison-Wesley, Boston, 2002.

[Coh03] Cohen, S.: *Predicting When Product Line Investment Pays.* Technischer Bericht CMU/SEI-2003-TN-017, Software Engineering Institute of the Carnegie Mellon University, Pittsburgh, 2003.

[DOO] *Telelogix DOORS/Analyst website.* http://www.telelogic.com.

[Fau01] Faulk, S.R.: *Product-Line Requirements Specification (PRS): an Approach and Case Study.* In: *Proceedings of the Fifth IEEE International Symposium on Requirements Engineering*, Seiten 48–55, Toronto, 2001.

[Har87] Harel, D.: *Statecharts: A Visual Formalism for Complex Systems.* Science of Computer Programming, 8:231–274, 1987.

[HMB02] Hardt, M., R. Mackenthun und J. Bielefeld: *Integrating ECUs in Vehicles - Requirements Engineering in Series Development.* In: *Proceedings of the IEEE Joint International Conference on Requirements Engineering*, Seiten 227–236, Essen, 2002.

[HN96] Harel, D. und A. Naamad: *The STATEMATE Semantics of Statecharts.* ACM Transactions on Software Engineering and Methodology, 5:4:293–333, 1996.

[HSVM00] Hein, A., M. Schlick und R. Vinga-Martins: *Applying Feature Models in Industrial Settings.* In: Donhoe, P. (Herausgeber): *Software Product Lines: Experiences and Research Directions*, Seiten 47–70, Norwell, 2000.

[JGJ97] Jacobson, I., M. Griss und P. Jonsson: *Software Reuse: Architecture, Process, and Organization for Business Success.* Addison-Wesley, ACM Press Books, Harlow, 1997.

[JM01] Juristo, N. und A.M. Moreno: *Basics of Software Engineering Experimentation.* Kluwer Academic Publishers, Dordrecht, 2001.

[KCH+90] Kang, K.C., S.G. Cohen, J.A. Hess, W.E. Novak und A.S. Petersen: *Feature-Oriented Domain Analysis (FODA) Feasibility Study.* Technischer Bericht CMU/SEI-90-TR-21, Software Engineering Institute of the Carnegie Mellon University, Pittsburgh, 1990.

[KS00] Kuusela, J. und J. Savolainen: *Requirements Engineering for Product Families.* In: *Proceedings of the 22nd International Conference on Software Engineering (ICSE)*, Seiten 61–69, Limerick, 2000.

[MJA+04] Muthig, D., I. John, M. Anastasopoulos, T. Forster, J. Dörr und K. Schmid: *GoPhone - A Software Product Line in the Mobile Phone Domain.* Technischer Bericht 025.04/E, Fraunhofer IESE, Kaiserslautern, 2004.

[Pre01] Prechelt, L.: *Kontrollierte Experimente in der Softwaretechnik: Potenzial und Methodik.* Springer-Verlag, Berlin, 2001.

[Rog95] Roger, E.K.: *Experimental Design: Procedures for the Behavioral Sciences.* Brooks/Cole Publishing Company, Pacific Grove, 1995.

[Sim] *MATLAB Simulink website.* http://www.mathworks.com.

[Sta] *I-Logix Statemate website.* http://www.ilogix.com.

[WL99] Weiss, D.M. und C.T.R. Lai: *Software Product-Line Engineering.* Addison-Wesley, Massachusetts, 1999.

A Anhang

Im Anhang sind einige der in diesem Experiment verwendeten Dokumente zusammengefasst:

1. Das Dokument für die Abfrage der Vorkenntnisse der Studierenden
2. Ein ausgefülltes Praktikumsberichtsheft einer Gruppe
3. Die Kernfunktionalität des verwendeten Bustürsteuergeräts
4. Der erste Kundenwunsch
5. Der zweite Kundenwunsch
6. Beispiel der Dokumentation der Produktlinie einer Gruppe

Vorkenntnisse

Name: _____

1. Hast du die Requirements Engineering Vorlesung besucht?

Ja ☐ Nein ☐

2. Hast du die Softwaretechnik Vorlesung besucht?

Ja ☐ Nein ☐

3. Wie beurteilst du deine Kenntnisse über (hierarchische) Zustandsautomaten?

- ☐ Ich kenne mich gut aus, habe bereits Erfahrungen damit
- ☐ Ich kenne diese Automaten (z.B. aus Vorlesungen) und ihre Semantik
- ☐ Ich habe noch Schwierigkeiten mit diesen Automaten
- ☐ Habe ich noch nie von gehört

4. Hast du schon einmal mit Modellierungswerkzeugen gearbeitet (z.B. Rational Rose, Together, usw.)?

- ☐ Ja, mit folgenden: _____
- ☐ Ich habe von folgenden Werkzeugen gehört (z.B. in Seminaren), aber noch nicht mit ihnen gearbeitet: _____
- ☐ Habe ich noch nie von gehört

5. Hast du schon einmal mit Modellierungswerkzeugen für eingebettete Systeme gearbeitet (z.B. Statemate, Stateflow, Artisan Realtimestudio, usw.)?

- ☐ Ja, mit folgenden: _____
- ☐ Ich habe von folgenden Werkzeugen gehört (z.B. in Seminaren), aber noch nicht mit ihnen gearbeitet: _____
- ☐ Habe ich noch nie von gehört

6. Hast du dich schon einmal mit der Programmierung eingebetteter Systeme beschäftigt (z.B. Vorlesung Echtzeitsysteme, Robotik-Labor, usw.)?

Ja ☐ Nein ☐

Berichtsheft

Gruppe: xyz

Mitglieder:

Kürzel:	Name:
tm	Student 1
ab	Student 2

Inhaltsverzeichnis

1. Allgemeines

Liebe Praktikumsteilnehmer,

dieses Berichtsheft wird Euch bis zum Ende des Praktikums begleiten. Bitte erfasst hier wöchentlich Eure Aufwände. Im letzten Abschnitt habt Ihr die Möglichkeit, Kritik und Anregungen, aber natürlich auch Lob festzuhalten über alles, was Euch im Lauf des Praktikums einfällt.

Viel Spaß und Erfolg
Euer Praktikumsteam

Das Praktikum läuft in drei Phasen ab:

- **Phase I**: Produktlinie entwickeln (Szenarien ausdenken)
- **Phase II**: Konkretes Produkt umsetzen
- **Phase III**: Qualitätssicherung

Für jede dieser Phasen habt ihr spezielle Abschnitte in diesem Dokument.

Im folgenden Abschnitt sind einige Modellierungs- und Dokumentationsrichtlinien enthalten. Bitte lest sie durch und haltet Euch daran.
Abschnitt 3 enthält die Formulare für die Aufwandserfassung, im letzten Abschnitt könnt Ihr eure Kritik loswerden.

2. Programmier- und Dokumentationsrichtlinien

In diesem Abschnitt findet Ihr hilfreiche Dokumente, die Euch während des Praktikumsverlaufs unterstützen sollen.
Bitte haltet Euch unbedingt an die im Anforderungsdokument der Kernfunktionalität (BSG) angegebenen Signal- und Datumsbezeichnungen und benennt neue (interne) von Euch eingeführte Signale mit folgendem Schema:

- Alle Events mit _E am Ende (z.B. TUER_1_MOTOR_OPEN_E)
- Alle Bedingungen mit _C am Ende (z.B. TUER_1_CLOSED_C)
- Alle Daten mit _D am Ende (z.B. FAHRZEUG_ZUENDUNG_D)

2.1 Phase I: Szenarien ausdenken

In dieser Phase sollt Ihr die Kernfunktionalität des BSGs mit dem Featurediagramm (FODA-Notation) und den Kontextdiagrammen strukturieren. Die Anforderungstabelle für die Kernfunktionalität müsst Ihr dann strukturiert nach dem Featurediagramm ausfüllen. Das Template für die Anforderungstabelle findet Ihr auf der Internetseite des Praktikums.

Für die FODA- Notation könnt Ihr unter
http://www.sei.cmu.edu/domain-engineering/FODA.html
detaillierte Informationen nachschlagen.

Parallel hierzu müsst Ihr die Kernfunktionalität in Statemate modellieren. Für die Kernfunktionalität solltet Ihr anschließend Testfälle in Form von *Simulation Control Programs* (SCPs) erstellen.
Bitte haltet Euch bei diesen SCPs an den im Folgenden dargestellten Beispiel-SCP und baut Eure Dokumente genauso auf.

```
PROGRAM  Normaler_Ablauf;

VARIABLE
  BOOLEAN Ring_E_ausgeloest; // merkt sich, dass Glocke ausgelöst wurde
  BOOLEAN aus; // merkt sich, dass Heizspule aus ist
  INTEGER anfang; // Zeitpunkt an dem START_E_ ausgelöst wird

INIT
  RESTORE_STATUS 'ANFANG';
END INIT;


// Event RING_E abfangen
// sollte nur 52s nach dem Start auftreten!
SET BREAKPOINT RING_E DO
  write('RING_E ausgeloest zum Zeitpunkt: ',cur_clock,'\n');
  IF ((cur_clock - anfang) == 52) THEN
    tr!(Ring_E_ausgeloest);
  END IF;
END BREAKPOINT;

// Event SET_OFF_E abfangen
// soll 52s nach dem Start auftreten!
SET BREAKPOINT SET_OFF_E DO
  IF ((cur_clock - anfang) == 52) THEN
    tr!(aus);
  END IF;
END BREAKPOINT;
```

```
BEGIN
  WRITE('\n\n\n');
  WRITE('================================\n');
  WRITE('  Testfall 1: normaler Ablauf    \n');
  WRITE('================================\n');

  // ----------------------------
  // Vorbedingungen
  // ----------------------------
  // Anfangsbedingungen wieder hergestellt

  // ----------------------------
  // Ablauf
  // ----------------------------
  // Leistung HIGH einstellen
  HIGH_E_;
  GO REPEAT;

  // Uhrzeit 5, 2 eingeben
  DIGIT5_E_;
  GO REPEAT;

  DIGIT2_E_;
  GO REPEAT;

  // START drücken
  START_E_;
  GO REPEAT;

  // Tür schließen
  fs!(DOOR_OPEN_C_);
  GO REPEAT;

  Anfang = cur_clock;

  // Zeit (52s) abwarten
  GO ADVANCE 52;

  GO REPEAT;

  // ----------------------------
  // Nachbedingungen
  // ----------------------------
  // RING_E Event ausgeloest
  IF (RING_E_ausgeloest == false) THEN
    write('ERROR 1.0 - RING_E nicht ausgeloest!\n');
  END IF;

  // Minutenanzeige auf 0
  IF (DISP_MIN_D != 0) THEN
    write('ERROR 1.1 - Minuten ungleich 0\n');
  END IF;

  // Sekundenanzeige auf 0
  IF (DISP_SEC_D != 0) THEN
    write('ERROR 1.2 - Sekunden ungleich 0\n');
  END IF;

  // Innenlicht aus
  IF (LIGHT_C) THEN
    write('ERROR 1.3 - Innenlicht brennt\n');
  END IF;

  // Heizspule auf Aus
  IF (not aus) THEN
    write('ERROR 1.4 - Heizspule nicht auf OFF\n');
  END IF;

  // Anzeige auf "OFF"
  IF (POWER_LIGHT_D != {AUS}) THEN
    write('ERROR 1.5 - Heizspule nicht auf AUS\n');
  END IF;

  STOP_SCP;
END;

END.
```

Zuletzt müsst Ihr Euch zukünftige Szenarien ausdenken und die erweiterte Funktionalität in Form von Variationspunkten

- im Featurediagramm
- in der Anforderungstabelle
- und im Statematemodell

darstellen.

Vergesst bitte nicht, Eure Aufwände wöchentlich nach Personen getrennt zu erfassen und an jens.kohlmeyer@informatik.uni-ulm.de zu schicken (siehe auch Abschnitte *3. Aufwandserfassung* und *3.1 Phase I – Szenarien ausdenken*).

2.2 Phase II: Produktumsetzung

Für die beiden **Produktliniengruppen 1** und **2** gilt:
Ihr erhaltet am 16.06 von uns einen konkreten Produktwunsch. Ihr habt dann eine Woche Zeit, um für dieses konkrete Produkt Euer Featurediagramm, die Anforderungstabelle und evtl. die Kontextdiagramme anzupassen.

Diese Dokumente werden dann mit einer anderen Gruppe getauscht und Ihr habt zwei Wochen Zeit die Anforderungen der anderen Gruppe in Eurem Modell umzusetzen, das konkrete Produkt also in Statemate zu modellieren.

Für die **Referenzgruppe**:

Ihr erhaltet am 16.06 zwei Produktwünsche von uns, und habt dann drei Wochen Zeit diese Anforderungen in Eure Diagramme einzuarbeiten und anschließend diese konkreten Produkte in Statemate zu realisieren.

2.3 Phase III: Qualitätssicherung

Für die beiden **Produktliniengruppen 1** und **2** gilt:
In dieser Phase erhaltet Ihr das fertige Produkt der anderen Gruppe mit Euren umgesetzten Anforderungen zurück und müsst dieses nun Qualitätssichern.

Für die **Referenzgruppe**:
Ihr sollt in dieser Phase Eure zwei Produkte mit Hilfe Eurer Testfälle abnehmen.

Hierfür erhaltet Ihr zu gegebener Zeit von uns entsprechende Abnahmeprotokolle.

3. Aufwandserfassung

Die aus der Einführungsphase bereits bekannt Aufwandserfassung wird fortgesetzt. Das Praktikum läuft in drei Phasen ab:

- **Phase I**: Produktlinie entwickeln (Szenarien ausdenken)
- **Phase II**: Konkretes Produkt umsetzen
- **Phase III**: Qualitätssicherung

Für jede Phase gibt es ein Formular für die Aufwandserfassung mit unterschiedlichen Aufwänden, die zu erfassen sind.

Bitte füllt das entsprechende Formular elektronisch jede Woche aus und sendet das **komplette Berichtsheft** an jens.kohlmeyer@informatik.uni-ulm.de.

3.1 Phase I – Szenarien ausdenken

Für diese erste Phase habt ihr vier Wochen Zeit.
Bitte klassifiziert Euren Aufwand **nach Person getrennt** nach folgenden **Klassen**:

- **A1**: Aufwand für Verständnis (z.B. Lesen der Anforderungen), Überlegungen, Kommunikation innerhalb der Gruppe
- **A2**: Aufwand für Kommunikation über Gruppengrenze hinaus (z.B. Betreuer)

- **B1**: Aufwand für Neumodellierungen in Statemate
- **B2**: Aufwand für Änderungen / Remodellierung in Statemate

- **C1**: Aufwand zur Erstellung der High-Level Testfälle
- **C2**: Aufwand zur Erstellung von Low-Level Testfällen

- **D**: Aufwand für Simulation und Testen (Kernfunktionalität)

- **E1**: Aufwand für Szenario (Überlegen, Umsetzen)
- **E2**: Aufwand für Variationspunkte (VPs)

- **F**: Aufwand für Dokumentationserstellung (Featurediagramme, Kontextdiagramme, Anforderungstabelle)

Wenn Ihr an Eurer Produktlinie arbeitet, so spezifiziert im Formular bitte, an welchem Szenario oder Variation Point ihr gerade arbeitet. Hierzu könnt Ihr im Feld „Tätigkeit“ kurz beschreiben, an welchem Szenario Ihr gerade arbeitet und welche Variation Points Ihr momentan gestaltet, und darauf dann in der Tabelle verweisen.

Woche vom **20.05.04 bis 26.05.04**

Tätigkeit:
In der Woche vom 20. – 26. Mai 2004 haben wir die Anforderungen gelesen, das Featurediagramm erstellt und ein paar Kontextdiagramme erstellt.

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Lesen der Anforderungen	A1	ab	1,5
2	Erstellen des Featurediagramms	F	ab	1
3	Erstellen des Featurediagramms mit Visio und Überarbeitung	F	ab	1
4	Lesen der Anforderungen	A1	tm	1,5
5	Erstellen des Featurediagramms	F	tm	1,5
6	Diskussion über das Featurediagramm	A1	tm/ab	2
7	Überarbeitung Featurediagramm, Erstellung Kontextdiagramme	F	tm/ab	3

Woche vom **27.05.04 bis 02.06.04**

Tätigkeit:
In der Woche vom 27.05.04 bis 02.06.04 haben wir das Featurediagramm überarbeitet, die Kontextdiagramme erstellt bzw. überarbeitet und die Anforderungstabelle in Excel geschrieben.

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Kontextdiagramme, Anforderungstabelle	F	Tm	3
2	Kontextdiagramme, Anforderungstabelle	F	Ab	3,5
3	Besprechung mit Betreuer	A2	Tm/ab	1
4	Activity Chart(s) in Statemate	B1	Tm/ab	2
5	Diskussion und Überlegungen über Modellierung in Statemate	A1	Tm/ab	2

Woche vom **03.06.04 bis 09.06.04**

Tätigkeit:
In der Woche vom 03.-09.06.04 haben wir die Kernfunktionalität des Bussteuergeräts in Statemate modelliert und Variationspunkte überlegt.

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Modulierung der Steuerung der Türen in Statemate	B1	Ab	3
2	Modellierung der Steuerung der Klimaanlage in Statemate	B1	Ab	0,5
3	Modellierung der Steuerung der Dachluken in Statemate	B1	Ab	2,5
4	Modellierung der Steuerung des Fahrersitzes in Statemate	B1	Ab	3
5	Variationspunkte überlegen (Türen, Dachluken, Innenbeleuchtung, Fahrersitz, Klimaanlage)	E2	Ab	1,0
6	Variationspunkte in Anforderungstabelle geschrieben	F	ab	0,5
7	Modellierung Aussenspiegel, Fenster, Diagnoseeinheit	B1	Tm	4,5
8	Variationspunkte überlegen (Aussenspiegel, Fenster, Diagnoseeinheit)	E2	tm	1
9	Besprechung des bisherigen Stands und Arbeitsaufteilung	A1	Ab/tm	2

Woche vom **10.06.04 bis 16.06.04**

Tätigkeit:
In der Woche vom 03.-09.06.04 haben wir die Kernfunktionalität des Bussteuergeräts in Statemate (re)modelliert, Variationspunkte modelliert und teilweise Tests durchgeführt.

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Remodellierung in Statemate	B2	tm	2,5
2	Testen und Simulation	D	tm	2
3	Modellieren Variationspunkte in Statemate	E2	tm	1
4	Besprechung der Modelle	A1	Ab/tm	1
5	Modellieren Variationspunkte in Statemate	E2	Ab	1
6	Remodellieren der Dachluken	B2	Ab	3
7	Featurediagramm mit Variationspunkten erweitert	F	Ab	0,5
8	Testen und Simulation	D	Ab	2
Verlängerung				
1	Testen der Kernfunktionen	D	tm	1
2	Erstellung der Systemabnahme-Protokolle	C1	Tm	2,5
3	Remodellierung bzw. Ausbesserung von Fehlern	B2	Tm	2
4	Testen der Kernfunktionen	D	ab	1,5
5	Erstellung der Systemabnahme-Protokolle	C1	ab	2
6	Remodellierung bzw. Ausbesserung von Fehlern	B2	ab	1

3.2 Phase II – Produktumsetzung - Realisierung

In dieser Phase müsst Ihr ein konkretes Produkt aus unseren Anforderungen und aus Eurer Produktlinie erstellen.
In der ersten Woche leitet Ihr eine Spezifikation des Produkts aus Eurer Produktlinie ab. Diese Spezifikation erhält die andere Gruppe.
In den folgenden zwei Wochen habt Ihr dann Zeit, den Produktwunsch der anderen Gruppe umzusetzen.
Bitte klassifiziert Euren Aufwand **nach Person getrennt** nach folgenden **Klassen**:

- **A1**: Aufwand für Verständnis (z.B. Lesen der Anforderungen), Überlegungen, Kommunikation innerhalb der Gruppe
- **A2**: Aufwand für Kommunikation über Gruppengrenze hinaus (z.B. Betreuer, andere Gruppe = Auftraggeber)

- **B1**: Aufwand für Ausgestaltung eines Variationspunktes in Statemate
- **B2**: Aufwand für Änderungen / Remodellierung am Kern in Statemate
- **B3**: Aufwand für zusätzlich notwendige Modellierung (ohne Variationspunktbezug)

- **C1**: Aufwand zur Anpassung der High-Level Testfälle
- **C2**: Aufwand zur Anpassung von Low-Level Testfällen

- **D**: Aufwand für Simulation und Testen (Kernfunktionalität)

- **F**: Aufwand für Dokumentationsanpassung (Featurediagramme, Kontextdiagramme, Anforderungstabelle)

Bitte erfasst Euren Aufwand **immer** mit Bezug auf einen Variationspunkt, sofern einer betroffen ist und gebt an, **welchen** VP Ihr gerade bearbeitet habt.

Woche vom **17.06.04 bis 23.06.04**

Tätigkeit:

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Durchlesen und Besprechung der zusätzlichen Anforderungen	A1	Ab/tm	1
2	Anpassen des Featurediagramms	F	Ab/tm	1
3	Anpassen der Kontextdiagramme	F	Tm	1,5
4	Anpassen der Anforderungstabelle	F	ab	1,5

Woche vom **24.06.04 bis 30.06.04**

Tätigkeit:

Schreiben der Testfälle für die Produktspezifischen Anforderungen

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Schreiben und Anpassen der Testfälle für die Produktspezifischen Anforderungen	C1	Ab/tm	3
2	Kommunikation innerhalb der Gruppe	A1	Ab/tm	1

Woche vom **01.07.04 bis 07.07.04**

Tätigkeit:

In der Woche vom 1.7. – 7.7. haben wir begonnen den Produktwunsch 2 umzusetzen.

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Durchlesen des Produktwunschs und Überlegungen	A1	tm	1
2	Durchlesen des Produktwunschs und Überlegungen	A1	ab	1
3	Modellieren in Statemate	B1	Tm	1
4	Modellieren der Klimaautomatik	B1	Ab	1,5
5	Modellieren der autom. Schließfunktion der Dachluken (zusätzliche Übergänge in die Kernfunktionalität aufgenommen)	B2	Ab	0,5
6	Modellierung der Spiegelheizung	B1	ab	0,25
7	Testen der Schließfunktion der Dachluken	D	Ab	0,5
8	Testen der Klimaautomatik	D	ab	0,25

Woche vom **08.07.04 bis 14.07.04**

Tätigkeit:

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Fehler in Statechart für Fenster in Fahrertür behoben	B2	Ab	0,5
2	VP „Komfortfunktion für die Fensterbedienung“ modelliert	B1	Ab	0,25
3	Integration des Hardwarepanels	B3 ?	ab	2,5
4	Benutzermanagement	B1	tm	5
5	Testen Benutzermanagement	D	Tm	3
6	Testen „Komfortfunktion für die Fensterbedienung“	D	Ab	0,5

3.3. Phase III - Qualitätssicherung

In dieser Phase muss das gelieferte Produkt mit den High- und Low-Level Testfällen getestet werden.
Auch hierbei bitten wir Euch Eure Aufwände getrennt nach Person und pro Woche zu erfassen. Bitte benutzt folgendes Klassifikationsschema.

- **A1**: Aufwand für Verständnis (z.B. Lesen der Anforderungen), Überlegungen, Kommunikation innerhalb der Gruppe
- **A2**: Aufwand für Kommunikation über Gruppengrenze hinaus (z.B. Betreuer)

- **C1**: Aufwand zur Erstellung der High-Level Testfälle

- **F**: Aufwand für Dokumentationserstellung (Abnahmeprotokoll)

- **Q**: Aufwand für das Testen des Produkts mit High-Level Testfällen

Woche vom **15.07.04 bis 21.07.04**

Tätigkeit:

Nr.	Beschreibung	Klasse	Person Kürzel	Dauer in Ph
1	Fehler in Modellierung behoben	B2	ab	2
2	Fehler in Modellierung behoben	B2	Tm	2
3	Probleme mit Statemate gelöst	A2	ab	0,5
4	Testen mit Panel	Q	Ab/tm	2

4. Anregungen und Kritik

In diesem Abschnitt habt Ihr Gelegenheit, Kritik oder Anregungen, die Euch während des Praktikumsverlauf auf- bzw. einfallen, zu notieren.
Haltet Euch nicht zurück, wir wollen schließlich das Praktikum in Zukunft weiter verbessern, und hierzu soll Eure Kritik beitragen.

Siehe separate E-Mail

Kernfunktionalität eines Bussteuergeräts

In einem modernen Bus werden viele Aufgaben von so genannten Steuergeräten übernommen. Solche Steuergeräte (SG) übernehmen beispielsweise die Steuerung der Türen oder des Motors.
In diesem Dokument werden die Anforderungen für ein Bussteuergerät (BSG) für einen modernen Bus beschrieben. Das BSG übernimmt Funktionen wie zum Beispiel die Steuerung der Türen, der Klimaanlage und die Einstellung des Fahrersitzes.

Im ersten Abschnitt wird eine Übersicht über die einzelnen Funktionalitäten des BSG gegeben und das Steuergerät in das Gesamtkonzept eingeordnet.
Im folgenden Abschnitt wird die zugrunde liegende Hardware beschrieben und die damit verbundenen Rahmenbedingungen für das BSG.
Der Abschnitt 3 enthält die ausführlichen Beschreibungen für die Einzelfunktionalitäten des BSG mit deren speziellen Anforderungen.

Inhaltsverzeichnis

1. Überblick über das BSG

In diesem Abschnitt wird ein Überblick über die Funktionalität und die Eigenschaften des BSG gegeben.

Das BSG ist im Fahrzeug integriert und kommuniziert mit den entsprechenden Sensoren und Stellmotoren über einen CAN-Bus. Seine Befehle vom Fahrer erhält das BSG zum einen von Bedienkonsolen im Armaturenbrett und zum anderen von entsprechenden Schaltern an den Bedienelementen.
Abbildung 1.1 enthält eine schematische Übersicht des BSG mit seinen peripheren Einheiten. Die grau hinterlegten Einheiten bezeichnen die Komponenten, die das BSG direkt ansteuert. Mit den weiß hinterlegten Komponenten (dies sind selbst Steuergeräte) findet eine Kommunikation über den CAN-Bus statt.

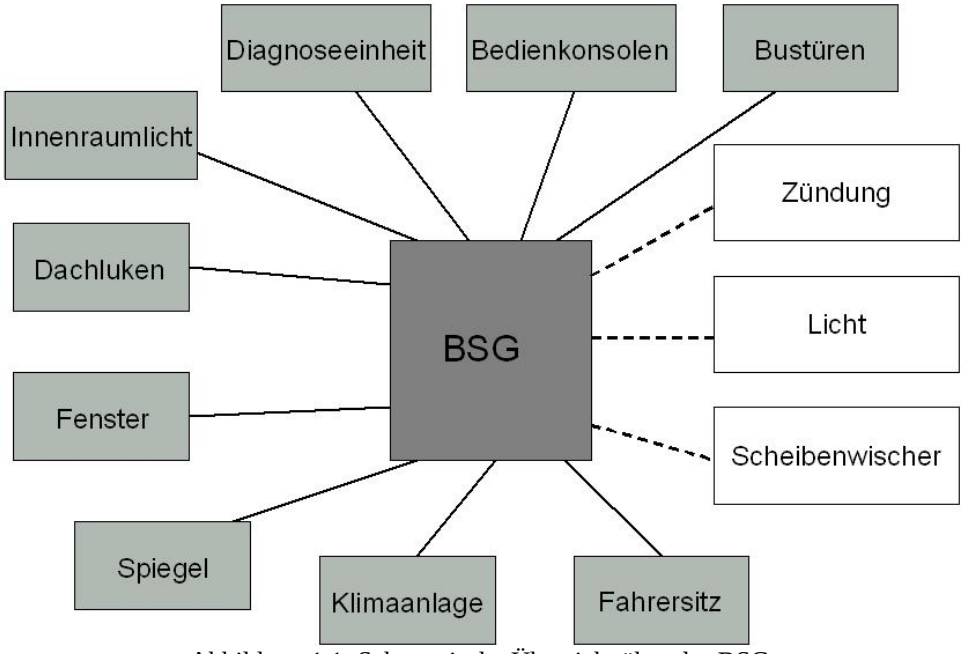


Abbildung 1.1: Schematische Übersicht über das BSG

1.1 Steuerung der Bustüren

Per Knopfdruck vom Fahrer an der Konsole werden die Türen einzeln im Fahrzeug geöffnet und geschlossen. Im Bus können bis zu drei Türen eingebaut werden. Der Fahrer muss an der Bedienkonsole erkennen, ob eine Tür geschlossen oder geöffnet ist.
Das Öffnen der Türen darf nur funktionieren, wenn der Bus steht und der Fahrer den Haltestellenmodus einstellt. Der Haltestellenmodus löst automatisch die Feststellbremse des Fahrzeugs aus und verhindert ein Rollen bzw. Fahren des Busses.
Zu beachten ist, dass die Fahrertür auch von Außen durch den Fahrer geöffnet werden kann. Hierzu wird der Schlüssel eingesteckt und eine Taste neben der Tür gedrückt. Die Tür wird dadurch entriegelt und geöffnet.
Die genaue Beschreibung der Funktionalität der Türen ist im Abschnitt *3.1 Steuerung der Bustüren* dargelegt.

1.2 Steuerung der Klimaanlage

Die Klimaanlage muss vom Fahrer aus gesteuert werden können. Im Fahrzeug wird ein Kühl- und Heizungssystem integriert, welches entsprechend den Innenraum beheizen oder kühlen kann. Beim Fahrer ist ein Bedienfeld, an welchem er die Intensität des Gebläses und die Kühl- bzw. Heizstufe einstellen kann.
Das Gebläse hat drei Einsatzstufen, von „stark“ über „mittel“ bis „schwach“. Nach Anschalten des Gebläses und Einstellen der Temperatur mit den Regler durch den Fahrer wird der Fahrzeuginnenraum beheizt oder gekühlt.
Für die genaue Beschreibung der Klimaanlage bitte unter Abschnitt *3.2 Steuerung der Klimaanlage* auf Seite 9 nachsehen.

1.3 Steuerung des Fahrzeuginnenraumlichts

Über die Konsole kann der Fahrer das Licht im Fahrzeuginnenraum an- und ausschalten.
Hierbei wird zwischen dem Licht vorne beim Fahrer an der Eingangstür und dem Licht im restlichen Fahrzeug unterschieden.
Die näheren Einzelheiten sind in Abschnitt *3.3 Steuerung des Fahrzeuginnenraumlichts* beschrieben.

1.4 Steuerung der Dachluken

Im Fahrzeug sind bis zu drei Dachluken eingebaut. Diese dienen im Notfall bei beispielsweise einem Umfallen des Fahrzeugs auch als Notausgänge, in dem die Dachluken mechanisch vom Fahrzeugdach getrennt werden können. Die Luken haben vorne und hinten in Fahrtrichtung jeweils einen Elektromotor, so dass sie flexibel zu öffnen sind.
In früheren Busmodellen konnten diese Dachluken von Hand (und damit von Fahrgästen) bedient werden. In Zukunft soll die Ansteuerung dieser Luken durch das BSG erfolgen (siehe auch Abschnitt *3.4 Steuerung der Dachluken*). Hierzu stehen in der Bedienkonsole beim Fahrer Knöpfe zur Verfügung. Der Fahrer muss am Display erkennen können, in welchem Zustand sich die Dachluken befinden.

1.5 Fahrersitzeinstellungen

Der Fahrersitz muss per Knopfdruck in verschiedene Positionen gefahren werden können.
Die Sitzfläche kann nach oben bzw. unten und nach vorne und zurück gefahren werden. Ermöglicht wird dies durch Taster, die direkt am Sitz angebracht sind.
Der Fahrersitz fährt solange in eine gewünschte Richtung, wie ein Schalter gedrückt wird und die Endposition noch nicht erreicht ist.
Mit einem solchen Taster kann auch die Neigung der Lehne eingestellt werden. In Abschnitt *3.5 Fahrersitzeinstellungen* wird dies näher erläutert.

1.6 Einstellung der Außenspiegel

Der Fahrer soll die beiden Außenspiegel vorne am Fahrzeug über eine Bedienkonsole einstellen können. Eine Drehung ist um die Längs- und die Querachse des Spiegels möglich.
Dieser Sachverhalt wird in Abschnitt *3.6 Einstellung der Außenspiegel* detailliert beschrieben.

1.7 Bedienung des Fensters in der Fahrertür

In der Fahrertür des Busses kann das Fenster nach unten bzw. oben gefahren werden. Die Ansteuerung des Fensters erfolgt über einen in der Fahrertür eingelassenen Schalter mit drei Positionen. Solange dieser Taster gedrückt ist, bewegt sich die Scheibe in die entsprechende Richtung bis die Endposition erreicht ist. Wird der Taster losgelassen, so ist die Bewegung sofort zu stoppen.
Nähere Einzelheiten werden in Abschnitt *3.7 Bedienung des Fensters in der Fahrertür* erläutert.

1.8 Die Bedienkonsolen beim Fahrer

Vorne beim Fahrer befinden sich rechts und links neben dem eigentlichen Armaturenbrett mit Tacho und sonstigen für das Führen des Fahrzeugs notwendigen Bestandteilen zwei Bedienkonsolen, die „Konsole Links“ und „Konsole Rechts“ (siehe auch Abbildung 1.8.1).
Mit der *Konsole Links* werden die beiden Außenspiegel des Fahrzeugs sowie die Klimaanlage gesteuert, mit der *Konsole Rechts* werden die Innenraumbeleuchtung, die Dachluken, die Türen und der Haltestellenmodus eingestellt.
Das Fenster in der Fahrertür wird über einen Schalter, welcher in der Tür eingelassen ist, gesteuert, der Fahrersitz über Taster am Sitz.
Tritt in einer Funktion des BSG ein Fehler auf, so ist dies dem Fahrer im Display anzuzeigen (beachte hierbei auch Abschnitt *1.9 Die Diagnoseeinheit*).
Die ausführliche Beschreibung der Eigenschaften der Bedienkonsolen befinden sich in Abschnitt *3.8 Die Bedienkonsolen beim Fahrer*.

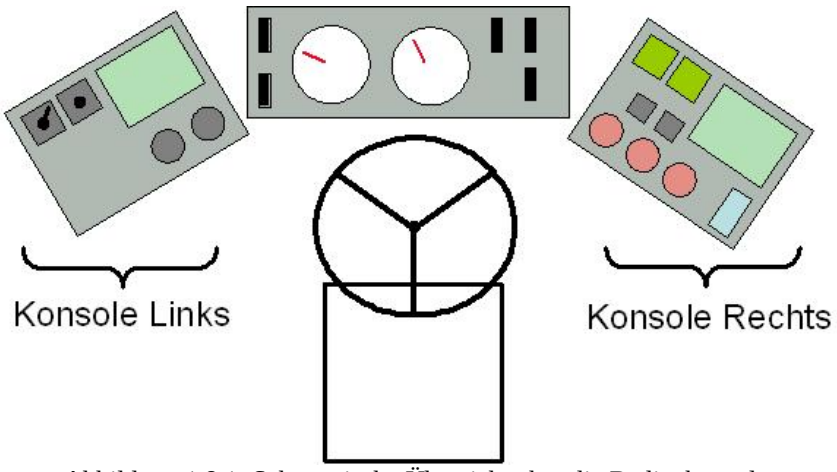


Abbildung 1.8.1: Schematische Übersicht über die Bedienkonsolen

1.9 Die Diagnoseeinheit

Die Diagnoseeinheit ist eine Funktion für die Wartung des Fahrzeugs. In einer Werkstatt haben Spezialisten die Möglichkeit, ein Diagnosegerät an das Fahrzeug anzuschließen und über dieses den Zustand der einzelnen Steuergeräte auszulesen.
Das BSG muss also Speicherplätze für die Fehlerfälle bereitstellen. Diese müssen ausgelesen werden können.
In Abschnitt *3.9 Die Diagnoseeinheit* auf Seite 20 wird dieser Sachverhalt näher beschrieben.

2. Hardware und Rahmenbedingungen

In diesem Abschnitt sind generelle Funktionen enthalten, die für die Funktionsweise des BSGs wichtig sind.
Alle in diesem Pflichtenheft erwähnten Bedingungen enden mit „_C“ (wie Condition), die Datenflüsse mit „_D“ und Events mit „_E“.
Das BSG arbeitet mit verschiedenen anderen Steuergeräten im Fahrzeug zusammen. Wichtig für das BSG ist u. a. der Zustand der Zündung. Dieser wird über den CAN-Bus zur Verfügung gestellt. Tabelle 2.1 enthält die Signale der Zündung, die über den CAN-Bus laufen und die für das BSG von Bedeutung sind.

Kürzel Zündung	Beschreibung
FAHRZEUG_ZUENDUNG_D	Dies ist ein Datum für die Fahrzeugzündung, also die Stellung des Schlüssels im Zündschloss. Folgende Werte können angenommen werden: - 0: kein Schlüssel im Zündschloss - 1: Schlüssel auf Stellung AUS - 2: Schlüssel auf Stellung BETRIEB / MOTOR AUS - 3: Schlüssel auf Stellung BETRIEB / MOTOR AN - 4: Schlüssel auf Stellung STARTEN
F_GESCHWINDIGKEIT_D	Über dieses Datum wird die Fahrzeuggeschwindigkeit über den CAN-Bus geschickt. Die Werte können zwischen 0 km/h und +150 km/h liegen.
AUSSEN_TEMPERATUR_D	Dieses Datum gibt die gemessene Außentemperatur über den CAN-Bus wieder. Der Temperaturbereich geht von -30 Grad Celsius bis +55 Grad Celsius.
INNEN_TEMPERATUR_D	Dieses Datum enthält die im Fahrzeuginnenraum gemessene Temperatur, deren Wertebereich von -20 Grad Celsius bis +65 Grad Celsius.

Tabelle 2.1 Zündungs- und Motorstatus

Neben der Zündung wird auch der Zustand des Fahrzeuglichts über den CAN-Bus geschickt. Mit „Fahrzeuglicht“ sind in diesem Zusammenhang die Frontscheinwerfer und die Rücklichter gemeint. Tabelle 2.2 enthält die Informationen über den Zustand des Fahrzeugslichts.

Kürzel Fahrzeuglicht	Beschreibung
FAHRZEUG_LICHT_D	Ein Datum welches anzeigt, wie der Status des Fahrzeuglichts ist. Folgende Werte können angenommen werden: - 0: Fahrzeuglicht aus - 1: Standlicht ist eingeschaltet - 2: Abblendlicht ist eingeschaltet - 3: Fernlicht ist eingeschaltet

Tabelle 2.2 Fahrzeuglichtstatus

Außerdem liegen auf dem CAN-Bus die in Tabelle 2.3 beschriebenen Bedingungen für den Zustand der Scheibenwischer an.

Kürzel Scheibenwischer	Beschreibung
SCHEIBENWISCHER_D	Ein Datum welches anzeigt, wie der Status des Scheibenwischers ist. Es können folgende Werte angenommen werden: - 0: Scheibenwischer aus - 1: Scheibenwischer Intervall an - 2: Scheibenwischer normal an - 3: Scheibenwischer schnell an

Tabelle 2.3 Scheibenwischerstatus

Die Anzahl der Türen und der Dachluken im Fahrzeug werden im Werk über Jumper am BSG eingestellt.
Das BSG ist solange eingeschaltet, so lange die Stromversorgung des Fahrzeugs angeschlossen ist.

3. Ausführliche Beschreibung der Funktionalitäten

In diesem Abschnitt werden die vom BSG gesteuerten Komponenten mit ihrer Funktionalität ausführlich beschrieben.

3.1 Steuerung der Bustüren

Der Bus verfügt über drei Doppeltüren, welche über einen komplizierten Mechanismus geöffnet und geschlossen werden können. Hierzu stehen dem Fahrer drei Knöpfe zur Verfügung, mit denen er die Türen einzeln steuern kann (siehe Abbildung 3.1.1). Die Türen werden ab jetzt von vorne nach hinten durchnummeriert, d.h. die Fahrertür ist Nummer 1, die hintere Tür die Nummer 3.

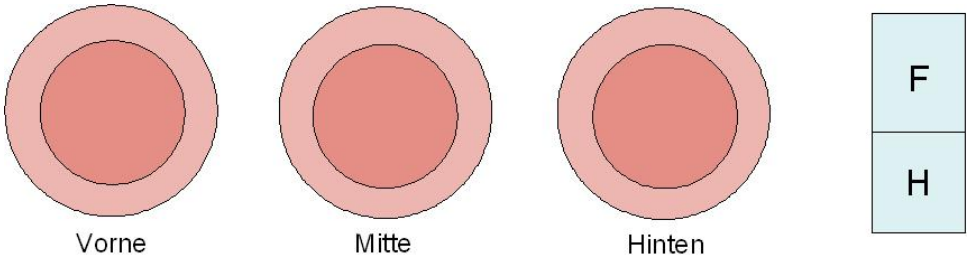


Abbildung 3.1.1: Bedienelement für den Haltestellenmodus und die Türen

Für jede Doppeltür gibt es einen Schalter (Vorne, Mitte und Hinten, also für die Tür 1, 2 und 3), welcher beim ersten Drücken die Tür öffnet, beim erneuten Drücken dann schließt. Jeder dieser Knöpfe beinhaltet ein Lämpchen, welches anzeigt, ob die Tür geöffnet (Lampe brennt) oder geschlossen (Lampe aus) ist.
Der Fahrer darf die Türen nur öffnen, wenn der Bus steht. Hierzu wird an einer Haltestelle der so genannte „Haltestellenmodus“ mit dem Kippschalter rechts in Abbildung 3.1.1 aktiviert. Der Modus kann nur aktiviert werden, wenn die Geschwindigkeit des Fahrzeugs weniger als 3 km/h beträgt. Nur in diesem Modus akzeptiert das BSG Befehle von den Türtastern. Will der Fahrer weiterfahren, so sind die Türen zu schließen und der „Fahrmodus“ mittels des Schalters einzustellen. Dies geht nur, wenn alle Türen geschlossen sind. Im Haltestellenmodus wird die Feststellbremse des Fahrzeugs angezogen, so wird ein Rollen bzw. Losfahren des Busses zuverlässig verhindert. Der Kippschalter steht entweder auf dem Modus „Haltestelle“ oder „Fahrt“.
Jeder der Türen ist mit einem Sensor verbunden, der anzeigt ob die Tür auf einen Widerstand beim Öffnen oder Schließen trifft. Ist dies der Fall, so ist die Bewegung sofort zu stoppen und die Tür muss in die entsprechende Gegenrichtung fahren, also z.B. beim Schließvorgang wieder aufgehen.
Die Tür 1 vorne beim Fahrer muss auch von Außen geöffnet werden können, damit das Fahrzeug auch durch den Fahrer betreten werden kann. Hierzu steckt er den Schlüssel in ein Schloss an der Tür 1 und betätigt kurz einen danebenliegenden Knopf. Dieser Knopf erzeugt ein Signal wie die Taste *Vorne* auf der *Konsole Rechts*.

Kürzel Türsteuerung	Beschreibung
HS_SWITCH_C	Gibt die Stellung des „Haltestellenkippschalters“ an. TRUE steht für „Haltestelle“, FALSE für „Fahrt“
FZG_TUER_TASTE_*x*_E	Das Event von der *Konsole Rechts*, welches beim

Kürzel Türsteuerung	Beschreibung
	Drücken auf die jeweilige Taste ausgelöst wird. *x* steht für 1, 2 oder 3.
FZG_TUER_LICHT_*x*_C	Diese Bedingung ist wahr, wenn das Lämpchen im Türtaster an ist, mit ihr wird das Lämpchen gesteuert. *x* steht wiederum für 1, 2 oder 3.
TUER_*x*_MOTOR_OPEN_E	Durch dieses Event wird der Motor der Tür *x* angehalten, die Tür *x* zu öffnen. Der Motor erkennt selbstständig, wann die Tür geöffnet ist und stoppt.
TUER_*x*_MOTOR_CLOSE_E	Analog zu oben weist dieses Signal den Motor der Tür *x* an, diese zu schließen. Wiederum erkennt der Motor, wann die Tür geschlossen ist und stoppt.
TUER_*x*_CLOSED_C	Ein Sensor reagiert, wann die Tür *x* komplett geschlossen ist. Dann wird die Bedingung für die Tür *x* auf wahr gesetzt. Sobald die Tür öffnet, wird die Bedingung auf falsch gesetzt. *x* steht wieder für 1,2 oder 3.
TUER_*x*_BLOCKED_ONCLOSE_C	Diese Bedingung wird wahr, wenn einer der Türflügel beim Schließen der Tür *x* auf einen Widerstand trifft. *x* für 1,2 oder 3.
TUER_*x*_BLOCKED_ONOPEN_C	Diese Bedingung wird wahr, wenn einer der Türflügel der Tür x beim Öffnen auf einen Widerstand trifft. *x* und ist ein Platzhalter analog wie oben.

Tabelle 3.3: Beschreibung der Signale für die Türsteuerung

3.2 Steuerung der Klimaanlage

Im Fahrzeug ist ein Kühl- bzw. Heizsystem eingebaut, mit welchem der Fahrer den Fahrzeuginnenraum Kühlen bzw. Heizen kann. Hierzu stehen ihm auf der *Konsole Links* die in Abbildung 3.2.1 dargestellten Knöpfe zur Verfügung.

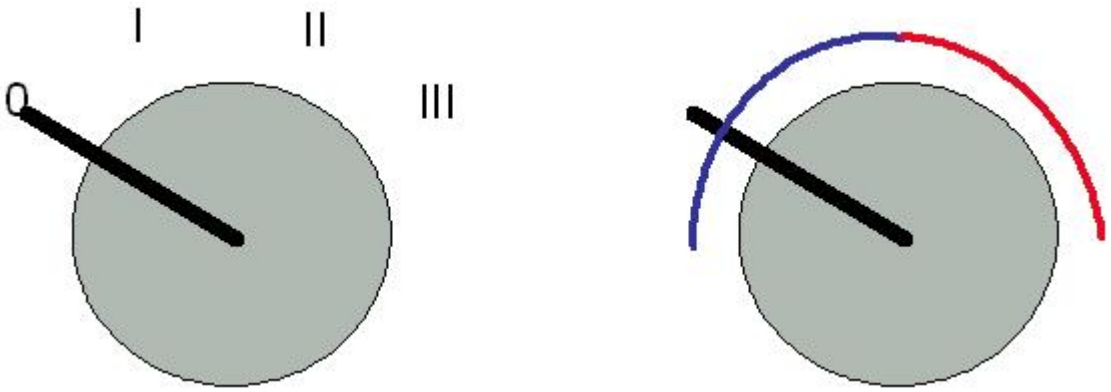


Abbildung 3.2.1: Bedienelement für die Klimaanlage.

Sowohl für das Heizen als auch das Kühlen stehen mit dem Gebläse drei Stufen zur Verfügung. Stufe I bezeichnet das schwache Arbeiten des Gebläses, Stufe II entspricht der mittleren Kraft, auf Stufe III arbeitet das Gebläse mit maximaler Kraft. Stufe 0 stellt das Gebläse ab. Das Gebläse wird mit dem linken Schalter bedient (siehe Abbildung 3.2.1). Die Heizung

bzw. Kühlung funktioniert nur auf den Stufen I bis III, auf Stufe 0 sind die Heizung bzw. die Kühlelemente ausgeschaltet.
Mit dem rechten Drehknopf wird vom Fahrer die gewünschte Temperatur der in das Fahrzeug strömenden Luft eingestellt. Steht der Schalter in Mittelstellung (also senkrecht), so kommt die Luft ohne zusätzliches Kühlen oder Erwärmen ins Fahrzeug, folglich mit der Umgebungstemperatur. Wird der Schalter nach links gedreht, so wird die Luft gekühlt, durch das Drehen nach rechts wird die Luft erwärmt. Im Knopf ist ein Rotationssensor, der die Stellung des Knopfes erfasst und auf den CAN-Bus legt (siehe Tabelle 3.3).

Kürzel Klimaanlage	Beschreibung
FZG_GEBLAESE_TASTE_*x*_C	Diese Bedingung zeigt an, in welcher Stellung der Gebläsetaster gerade steht. *x* steht hierbei für 0, 1, 2, oder 3, entsprechend der Gebläsestufe. Die Bedingung für die momentan eingestellte Stufe ist wahr.
FZG_KLIMA_TEMP_D	Dieses Datum gibt die Stellung des rechten Drehknopfs an. Der Wertebereich geht von 1 bis 99, wobei 50 den senkrechten Stand des Reglers angibt. Bei Werten von 1 bis 49 wird die Luft gekühlt, von 51 bis 99 wird geheizt.
FZG_GEBLAESE_POWER_*x*_E	Mit diesem Signal wird das Gebläse gesteuert. *x* steht für 1, 2 oder 3. Wird das Signal ausgelöst, so arbeitet das Gebläse mit der Stufe *x*.
FZG_GEBLAESE_STOP_E	Dieses Signal stoppt das Gebläse.

Tabelle 3.3: Beschreibung der Signale für die Klimaanlage

3.3 Steuerung des Fahrzeuginnenraumlichts

Der Innenraum des Fahrzeugs ist mit Lampen ausgestattet, um auch im Dunklen für genügend Sicherheit und Komfort im Fahrzeuginneren zu sorgen. Der Fahrer hat vorne bei sich am Türraum ein Licht, um bei Nacht zahlenden Kunden genügend Licht zu bieten, das von den anderen Lampen getrennt zu bedienen ist.

Abbildung 3.3.1: Knöpfe für das Fahrzeuginnenlicht

An der *Konsole Rechts* befinden sich zwei Taster, mit denen der Fahrer das Innenraumlicht und das „Fahrerlicht“ getrennt voneinander bedienen kann (siehe auch Abbildung 3.3.1). Die Taster haben zwei Positionen, „gedrückt“ und „normal“, die abwechselnd durchgeschaltet werden.
In Tabelle 3.3 werden die für die Steuerung des Fahrzeuginnenraumlichts benötigten Signale beschrieben.

Kürzel Fahrzeuglicht	Beschreibung

Kürzel Fahrzeuglicht	Beschreibung
FZG_INNENLICHT_TASTE_*x*_C	Diese Bedingung ist wahr, wenn die jeweilige Taste im Modus „gedrückt“ ist. *x* steht für FAHRER bzw. REST.

Tabelle 3.3: Beschreibung der Signale für das Fahrzeuglicht

3.4 Steuerung der Dachluken

In diesem Abschnitt wird die Steuerung der Dachluken in dem Bus von einer Bedienkonsole beim Fahrer aus beschrieben.
Die Dachluken dürfen nur angesteuert werden, wenn die Zündung auf Stellung BETRIEB steht.
Im Fahrzeug sind zwei Dachluken eingebaut. Diese dienen im Notfall bei beispielsweise einem Umfallen des Fahrzeugs auf die Seite auch als Notausgänge, indem die Passagiere die Luken komplett vom Fahrzeugdach abtrennen und durch das Loch ins Freie klettern können. Die Luken haben vorne und hinten in Fahrtrichtung jeweils einen Elektromotor, so dass sie flexibel zu öffnen sind (siehe Abbildung 3.4.1).
In früheren Busmodellen konnten diese Dachluken von Hand (und damit von Fahrgästen) bedient werden. In Zukunft soll die Ansteuerung dieser Luken durch das Bustürsteuergerät (BSG) erfolgen.
Der Fahrer kann über die Bedienkonsole folgende Modi einstellen:

- Geschlossen: Die Dachluke ist geschlossen (siehe Abbildung 3.4.1 A).
- Vorne Offen: Die Luke wird nur vorne geöffnet. Hierzu wird der vordere Elektromotor angesteuert. Dies führt zu einem starken Luftzug in das Fahrzeug (siehe Abbildung 3.4.1 B).
- Beide Offen: Die Luke wird vorne und hinten geöffnet, sie „schwebt“ praktisch parallel zum Fahrzeugdach (siehe Abbildung 3.4.1 C).
- Hinten Offen: Die Luke wird nur hinten geöffnet (siehe Abbildung 3.4.1 D).

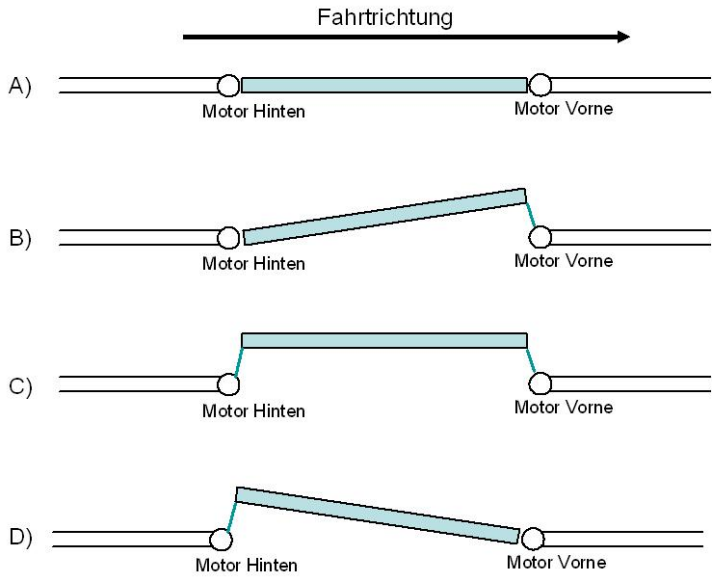


Abbildung 3.4.1: Schematischer Querschnitt durch das Fahrzeugdach quer in Fahrtrichtung

Aufbau des Mechanismus

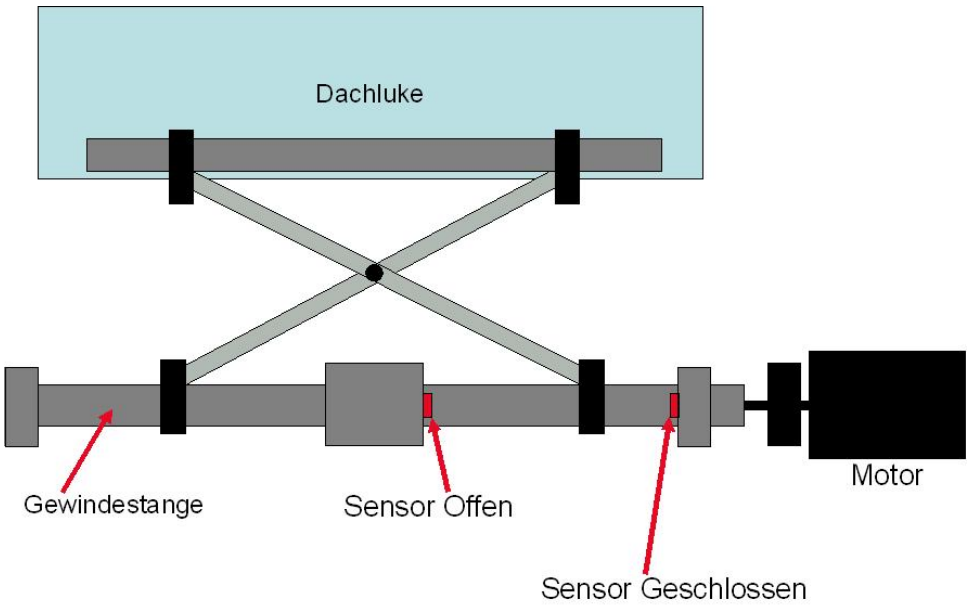


Abbildung 3.4.3 : Schließ- und Öffnungsmechanismus der Dachluken (Schematisch)

Ein Motor treibt eine Gewindestange an, auf der zwei Muttern laufen. Durch Drehung der Gewindestange werden die Muttern entweder vom Mittelpunkt weg geschoben, dann schließt sich die Luke, oder zum Mittelpunkt hingeschoben, entsprechend öffnet sich die Dachluke.
Zwei Sensoren (*Sensor Offen*, *Sensor Geschlossen*) zeigen an, wann die Luke auf dieser Seite vollständig geschlossen bzw. offen ist.
Zum Öffnen der Dachluke in diesem Bereich wird der Motor in die entsprechende Richtung gestartet, sobald der Sensor *Sensor Offen* anspricht wird der Motor gestoppt, die Dachluke ist offen. Umgekehrtes gilt für das Schließen.
Wird der Motor gestartet, aber nach drei Sekunden hat der entsprechende Sensor noch nicht reagiert, so ist von einem Fehler auszugehen.

An der Bedienkonsole vorne beim Fahrer befinden sich zwei Knöpfe für die Bedienung der Dachluken und ein Display, in welchem unter anderem der Öffnungszustand der Dachluken angezeigt wird (Abbildung 3.5.2, für das Display siehe Abbildung 3.8.2).

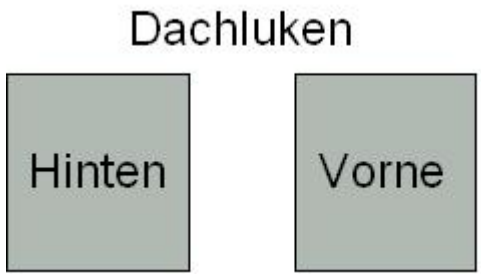


Abbildung 3.4.2: Bedienelement für die Steuerung der Dachluken

Mit den beiden Knöpfen wird der „Öffnungszustand“ für die beiden Dachluken zyklisch durchgeschaltet, und zwar in dieser Reihenfolge:

- Geschlossen (siehe Abbildung 3.4.1 A)

- Vorne offen (Abbildung 3.4.1 B)
- Beide offen (Abbildung 3.4.1 C)
- Hinten offen (Abbildung 3.4.1 D)

Mit jedem Knopfdruck des Fahrers wird die entsprechende Dachluke in die Position gefahren, die zyklisch an der Reihe ist. Ist also bspw. die vordere Luke geschlossen und der Fahrer drückt einmal auf den Knopf, so fährt die Luke in die Stellung „Vorne offen".
Bei der Bedienung durch den Fahrer sind folgende Funktionalitäten zu berücksichtigen:

- Im Display muss nach jedem Tastendruck durch den Fahrer sofort die neue Stellung der jeweiligen Dachluke angezeigt werden.
- Die Luke wird allerdings erst in diese Position gefahren, wenn der Fahrer mindestens eine Sekunde die Taste nicht mehr gedrückt hat. So wird vermieden, dass durch dreimaliges Drücken der Taste kurz hintereinander die Luke immer wieder neu angesteuert werden muss. Der Fahrer hat dadurch auch die Möglichkeit Schritte bei der Reihenfolge zu „überspringen" (Beispiel: die Luke ist auf „Beide Offen" und soll jetzt ganz geschlossen werden: der Fahrer drückt zweimal kurz die Taste, nach einer Sekunde wird die Luke in die zuletzt eingestellte Position gefahren).

Selbstverständlich sind gewisse Sicherheitsstandards einzuhalten. So muss bei Blockierung der Bewegung in eine bestimmte Richtung die Bewegung sofort gestoppt werden. Dies wird vom Motor übernommen, der bei zu großer Belastung sofort selbstständig stoppt und ein entsprechendes Signal an das Steuergerät sendet. Von einem Fehler ist auch auszugehen, wenn die Dachluke innerhalb von 3 Sekunden nicht auf einen Befehl des Fahrers reagiert, d.h. z.B. die Luke hinten auffahren soll, der entsprechende Sensor aber nach spätestens drei Sekunden nicht anspricht.
In der folgenden Tabelle werden alle Events, Bedingungen, Sensorsignale und Motoransteuerungen für das Steuergerät mit zwei Dachluken (die vordere wird ab sofort mit Luke 1, die hintere mit Luke 2 bezeichnet) spezifiziert.

Kürzel Motorsteuerung	Beschreibung
MOTOR_DL_*x*_*y*_*z*_E	Dies ist das Event für die Motorsteuerung an einer Dachluke. *x* steht hierbei für 1 oder 2 (Dachluke 1 oder 2), *y* für V (Vorne) oder H (Hinten) und *z* für ZU oder AUF. Das Signal **MOTOR_DL_1_V_AUF_E** bedeutet also, dass an den vorderen Motor der Dachluke 1 das Signal zum Aufmachen geschickt wird.
MOTOR_*x*_*y*_STOPPED_E	Der Motor schickt ein Signal, wenn er stoppt. Dies passiert z.B. wenn die Endposition an der Gewindewelle erreicht wird oder der Motor blockiert wird. *x* steht für 1 oder 2 (Luke 1 oder 2), *y* für V (Vorne) oder H (Hinten).
SENSOR_DL_*x*_*y*_*z*_C	Eine Bedingung für die Sensoren an den Dachluken. Sie wird wahr, wenn am Sensor ein Signal anliegt. *x* steht hierbei für 1 oder 2 (Dachluke 1 oder 2), *y* für V (Vorne) oder H (Hinten) und *z* für GESCHLOSSEN oder OFFEN. Wenn also **SENSOR_DL_1_V_OFFEN_C** wahr ist, so ist die Dachluke 1 Vorne offen.
Kürzel Display-Steuerung	Beschreibung
DL_*x*_BUTTON_E	Das Event für die Knöpfe an der Bedienkonsole zur Steuerung der Dachluken (zyklisch). *x* steht für 1 oder 2 (Luke 1 und Luke 2). **DL_1_BUTTON_EVENT** bedeutet, dass der Knopf für die Luke 1 gedrückt wurde.

Kürzel Motorsteuerung	Beschreibung
DL_*x*_STATUS_D	Hierüber wird an das Display der Status der einzelnen Dachluken ausgegeben, wobei *x* für 1 oder 2 (Dachluke 1 oder 2) steht. Folgende Werte kann der Status haben: - 0: Dachluke geschlossen - 1: Vorne Offen - 2: Beide Offen - 3: Hinten Offen
DL_*x*_ERROR_C	Diese Bedingung wird wahr, wenn an der Dachluke *x* ein Fehler festgestellt wurde.

Tabelle 3.4: Beschreibung der Signale für die Dachluken

3.5 Fahrersitzeinstellungen

Das BSG ist zuständig für die Kontrolle der elektrischen Fahrersitzeinstellung.
Abbildung 3.5.1 zeigt eine schematische Übersicht über den Fahrersitz und die Bedienelemente für die Einstellungen des Sitzes.
Der Fahrersitz darf nur eingestellt werden, wenn

- die Zündung auf Stellung BETRIEB steht
- die Geschwindigkeit des Fahrzeugs weniger als 10 km/h beträgt
- kein Taster der Außenspiegeleinstellung betätigt wird

Die Sitzfläche kann nach oben bzw. unten gefahren werden. Im Bedienelement erfolgt dies durch den mittleren Taster. Wird dieser nach oben gedrückt, so fährt der Sitz nach oben bis entweder der Taster losgelassen oder der Sitz in der maximalen Höhe angekommen ist, nach unten entsprechend. In Ruhestellung des Tasters (mittig) wird der Sitz nicht bewegt, der Taster kehrt nach dem Loslassen von selbst in die Ruhestellung zurück.
Die Sitzfläche kann ebenso nach vorne oder zurück gefahren werden. Dies ermöglicht der Taster ganz links im Bedienfeld, er funktioniert auf gleiche Art und Weise wie der Taster für die Höheneinstellung.
Mit dem Taster rechts kann die Neigung der Lehne eingestellt werden. Drückt man den Taster nach links, so wird die Neigung steiler, nach rechts wird die Lehne flacher. Auch diese Schalter funktioniert nach dem gleichen Prinzip wie der erste.
Auch hier sind wiederum Sicherheitsstandards zu garantieren. So ist bei Blockierung der „Fahrt“ des Sitzes die Bewegung sofort zu stoppen.

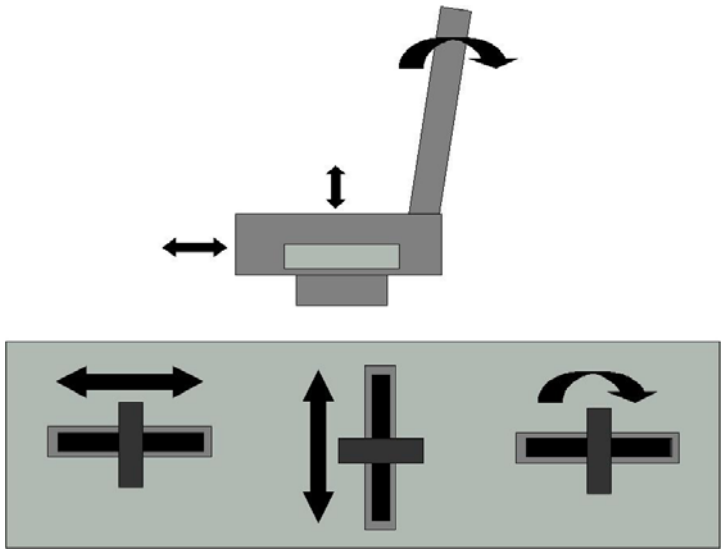


Abbildung 3.5.1: Schema des Fahrersitzes mit seitlich angebrachten Schaltern

Für die Ansteuerung des Fahrersitzes ist Folgendes zu beachten:

- Es darf nur eine Bewegungsrichtung des Sitzes angewählt werden. Das BSG fährt den Sitz bei Bedienung durch den Fahrer immer nur in maximal eine Richtung. Werden mehrere Tasten gleichzeitig gedrückt, so wird die zuletzt gedrückte Fahrtrichtung angesteuert.
- Die Motoren des Sitzes sind mit einem Überspannungsschutz ausgerüstet. Bei zu großer Belastung (z.B. bei einer Blockierung des Sitzes) stoppen die Motoren automatisch. Die Taster reagieren erst wieder auf ein Kommando, wenn sie zunächst alle losgelassen wurden.

Im Sitz sind Sensoren angebracht, welche die aktuelle Position des Fahrersitzes mit der Stellung der Rückenlehne angeben. Die gültigen Werte für die möglichen Sitzpositionen sind in Tabelle 3.5 beschrieben. Diese Werte dürfen nicht verletzt werden, hierfür hat das BSG Sorge zu tragen. Der Sitz bewegt sich in horizontaler und vertikaler Richtung mit etwa einem Zentimeter pro Sekunde, was einer Werteänderung von 50 entspricht.
In der Lehne ist ein Rotationssensor integriert, der die Stellung der Lehne in Grad angibt. 90 Grad ist die genau senkrechte Position der Lehne. Die Stellung der Lehne wird pro Sekunde um 5 Grad durch den Tastendruck verändert.

Kürzel Fahrersitz	Beschreibung
SITZ_TASTER_HOR_*x*_C	Eine Bedingung die anzeigt, ob der Taster für die horizontale Sitzbewegung gedrückt wird (also Bewegung nach vorne bzw. zurück). Ist wahr, wenn der Taster gedrückt wird. *x* steht für VOR und ZURUECK.
SITZ_TASTER_VER_*x*_C	Diese Bedingung zeigt an, ob der Taster für die vertikale Sitzbewegung gedrückt wird. In diesem Fall ist die Bedingung wahr. *x* steht für HOCH bzw. RUNTER.
SITZ_LEHNE_*x*_C	Zeigt an, ob der Taster für die Lehnenverstellung gerade gedrückt ist (dann wahr). *x* steht für FLACHER bzw. STEILER.
SITZ_MOTOR_*x_y*_E	Signal, um den entsprechenden Sitzmotor anzusteuern. *x* steht hierbei für

Kürzel Fahrersitz	Beschreibung
	- HOR (*y* entsprechend für VOR bzw. ZURUECK) - VER (*y* für HOCH bzw. RUNTER) - LEHNE (*y* für FLACHER bzw. STEILER)
SITZ_MOTOR_*x*_STOP_E	Dieses Signal stoppt den entsprechenden Motor. *x* steht für HOR (horizontal), VER (vertikal) oder LEHNE.
SITZ_MOTOR_*x*_STOPPED_E	Dieses Signal wird vom Motor an das BSG über den CAN-Bus geschickt, falls er wegen einer Überbelastung stoppt. Der Platzhalter *x* steht für HOR, VER bzw. LEHNE.
SITZ_POS _HOR_D	Ein Datum für die horizontale Sitzposition. Der gültige Wertebereich geht von 60-1010.
SITZ_POS_VER_D	Datum für die vertikale Sitzposition. Der Sensor liefert Werte, welche den Bereich zwischen 20 und 420 nicht verlassen dürfen.
SITZ_POS_LEHNE_D	Der Winkel der Lehne wird mit einem Rotationssensor abgegriffen. Die Werte müssen zwischen 90 und 160 Grad liegen.

Tabelle 3.5: Beschreibung der Signale für die Fahrersitzeinstellung

3.6 Einstellung der Außenspiegel

Mit dem BSG kann der Fahrer die Position der beiden Außenspiegel im Fahrzeug einstellen. Die Außenspiegel im Fahrzeug dürfen nur eingestellt werden (d.h. die Spiegelmotoren dürfen angesteuert werden), wenn

- die Zündung auf Stellung BETRIEB steht
- die Sitzverstellung nicht betätigt wird

Jeder der beiden Spiegel (zur Unterscheidung im Folgenden *Spiegel_Links* und *Spiegel_Rechts* in Fahrtrichtung genannt) kann in horizontaler und vertikaler Richtung eingestellt werden. Im Spiegelgehäuse befinden sich Sensoren, welche die momentane Stellung der Spiegelscheibe erfassen. Da keinerlei Endsensoren angebracht sind, dürfen die Werte der Spiegelsensoren einen bestimmten Wertebereich determiniert durch das Gehäuse nicht verlassen. Die Kontrolle ist Aufgabe des BSGs.
Für die Ansteuerung stehen dem Fahrer auf der *Konsole Links* die zwei in Abbildung 3.6.1 gezeigten Elemente zur Verfügung.

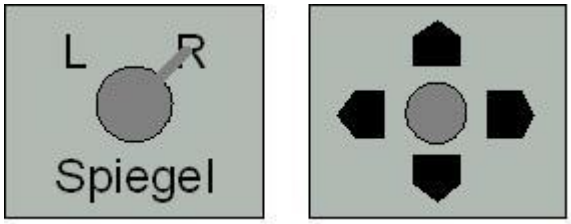


Abbildung 3.6.1: Bedienfelder für die Spiegeleinstellung

Mit dem Element auf der linken Seite wird eingestellt, ob der *Spiegel_Links* (Stellung auf L) oder der *Spiegel_Rechts* (Stellung auf R) angesteuert wird. Im rechten Element kann der Fahrer mittels einer Art kleiner Joystick den Spiegel in die gewünschte Richtung bewegen. Wird der Joystick zum Beispiel nach rechts gedrückt, so dreht sich die entsprechende Spiegelscheibe um die vertikale Achse nach rechts. Wird der kleine Joystick losgelassen, so kehrt er automatisch in die Ruhestellung in die Mitte zurück. Der Schalter erkennt nur eine Bewegungsrichtung. Der jeweilige Spiegel fährt nur, solange entweder der Joystick in die Richtung gedrückt wird oder der gültige Wertebereich nicht verlassen wird.
Der Wertebereich für die beiden Spiegel ist jeweils der Gleiche:

- Horizontal: -20° (Anschlag ganz links) bis +20° (Anschlag ganz rechts)
- Vertikal: -15° (Anschlag ganz unten) bis +15° (Anschlag ganz oben)

Pro Halbesekunde wird der Spiegel durch den Motor um +/-2° in vertikaler oder horizontaler Richtung bewegt.

Kürzel Außenspiegel	Beschreibung
SPIEGEL_ANSTEUERN_LINKS_C	Eine Bedingung, welcher Spiegel momentan eingestellt werden kann. Ist wahr, wenn Spiegel_Links angesteuert wird, falsch wenn der rechte Außenspiegel angesprochen wird (siehe Abbildung 3.7.1, linkes Element).
SPIEGEL_*y*_TASTER_C	Diese Bedingungen werden durch den kleinen Joystick im Bedienfeld auf der *Konsole Links* durch drücken des Joysticks in die jeweilige Richtung gesetzt. y steht hierbei für: - HOCH: Joystick wird nach oben gedrückt - RUNTER: Joystick wird nach unten gedrückt - RECHTS: Joystick wird nach rechts gedrückt - LINKS: Joystick wird nach links gedrückt
SPIEGEL_*x*_MOTOR_*y*_E	Die Signale für die Steuerung der Spiegelstellmotoren. *x* steht für LINKS und RECHTS, *y* kann folgende Werte annehmen: - HOCH: Motor dreht Spiegel nach oben - RUNTER: Motor dreht Spiegel nach unten - RECHTS: Motor dreht Spiegel nach rechts - LINKS: Motor dreht Spiegel nach links.
SPIEGEL_*x*_MOTOR_STOP_E	Signal, um den entsprechenden Spiegelmotor zu stoppen. *x* steht hierbei für LINKS bzw. RECHTS
SPIEGEL_*x*_*y*_POS_D	Dieses Datum enthält die aktuelle Position des Spiegels *x* in vertikaler bzw. horizontaler Richtung. *x* steht für LINKS bzw. RECHTS, *y* für HOR bzw. VERT.

Tabelle 3.6: Beschreibung der Signale für die Außenspiegel

Zu beachten ist, dass das BSG für die Positionen der Spiegel entsprechend Speicherstellen zur Verfügung stellt und die Werte fortlaufend überwacht.

3.7 Bedienung des Fensters in der Fahrertür

Aus Komfortgründen kann der Fahrer das Fenster in der Fahrertür elektrisch öffnen bzw. schließen. Die Scheibe ist hierbei in vertikaler Richtung mittels eines Elektromotors bewegbar.
In der Fahrertür befindet sich ein Taster für die Steuerung der Scheibe. In Ruhestellung wird die Scheibe nicht bewegt, drückt der Fahrer den Schalter nach vorne, so fährt die Scheibe nach oben, wird der Schalter nach hinten gedrückt, so senkt sich die Scheibe entsprechend. Wird der Taster losgelassen, so springt er automatisch in die Ruhestellung zurück.

Solange der Fahrer eine Fensterheber-Taste drückt, fährt die Scheibe in die zugehörige Richtung, bis eine der Endpositionen (*Oben* bzw. *Unten*) erreicht ist. Falls die Scheibe blockiert, ist die Bewegung sofort zu stoppen und muss umgekehrt werden, d.h. die Scheibe fährt entsprechend in die andere Position. Die Blockierung kann anhand des Scheibenmotors festgestellt werden. Die Endposition *Unten* wird mittels eines Sensors festgestellt, die Endposition *Oben* mittels einer Widerstandsfolie erkannt.
Eine Kontrolle von Fehlern findet bis auf die Widerstandskontrolle nicht statt.

Kürzel Fensterheber	Beschreibung
FENSTER_MOTOR_*x*_E	Ein Event für den Motor des Fensterhebers. *x* steht hierbei für AUF oder ZU, je nachdem in welche Richtung (oben bzw. unten) die Scheibe gefahren werden soll. Durch das Auslösen des Events wird der Motor entsprechend angesteuert.
FENSTER_MOTOR_STOP_E	Wird dieses Event ausgelöst, so wird der Motor des Fensterhebers gestoppt.
FENSTER_MOTOR_STOPPED_E	Der Motor des Fensterhebers schickt ein Signal, wenn er stoppt, ohne dass vorher das Signal FENSTER_MOTOR_STOP_E ausgelöst wurde (d.h. der Motor stoppt aus außergewöhnlichen Gründen). Dies ist zum Beispiel der Fall, wenn er wegen eines zu großen Widerstands die Bewegung aus Schutzgründen stoppt.
FENSTER_*x*_C	Diese Bedingung zeigt an, ob das Fenster ganz oben (*x* = OBEN) bzw. unten (*x* = UNTEN) ist. Sie wird entweder durch den Sensor unten in der Fahrertür oder durch die Widerstandsfolie auf wahr gesetzt.
FENSTER_TASTER_*x*_C	Diese Bedingung zeigt an, wenn der Taster des Fensterhebermechanismus in eine bestimmte Richtung gedrückt wird. *x* steht hierbei für HOCH oder RUNTER.

Tabelle 3.7: Beschreibung der Signale für den Fensterheber

3.8 Die Bedienkonsolen beim Fahrer

Beim Fahrer sind links und rechts neben dem eigentlichen Armaturenbrett zwei Bedienkonsolen angebracht, die *Konsole Links* und die *Konsole Rechts*.
Abbildung 3.8.1 zeigt die *Konsole Links*.

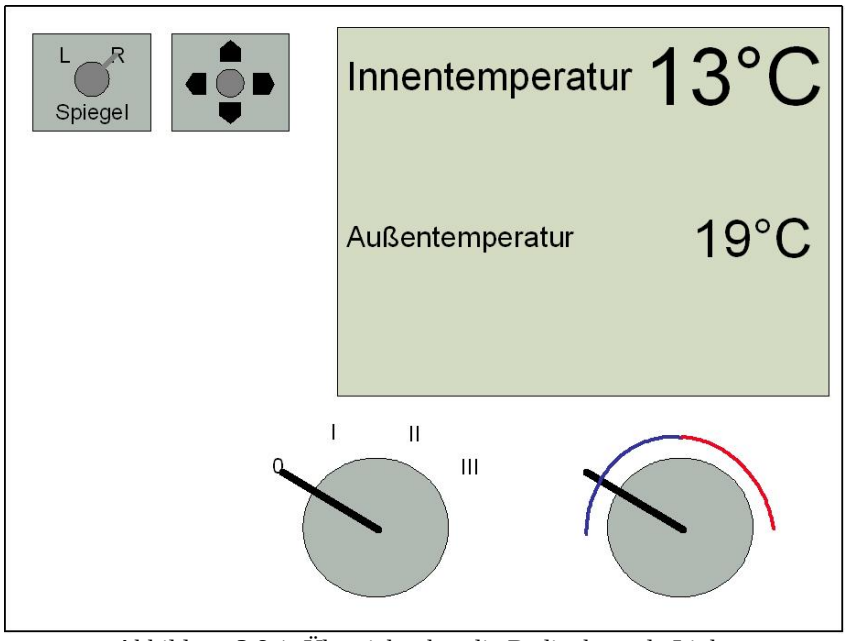


Abbildung 3.8.1: Übersicht über die Bedienkonsole Links

Mit dieser Konsole kann der Fahrer die Außenspiegel des Fahrzeugs einstellen. Hierzu befinden sich oben links die entsprechenden Bedienelemente. Näheres hierzu siehe Abschnitt *3.6 Einstellung der Außenspiegel*.
Rechts unten befindet sich die Einheit für die Steuerung der Klimaanlage bzw. Heizung, welche in Abschnitt *3.2 Steuerung der Klimaanlage* detailliert dargestellt wird.
Das Display oben rechts enthält Informationen für den Fahrer hinsichtlich der Temperatur im Fahrzeuginneren und –äußeren Etwaige Fehlermeldungen die Spiegel oder die Klimaanlage bzw. Heizung betreffend werden hier angezeigt.

Abbildung 3.8.2 enthält die Übersicht über die *Konsole Rechts*.

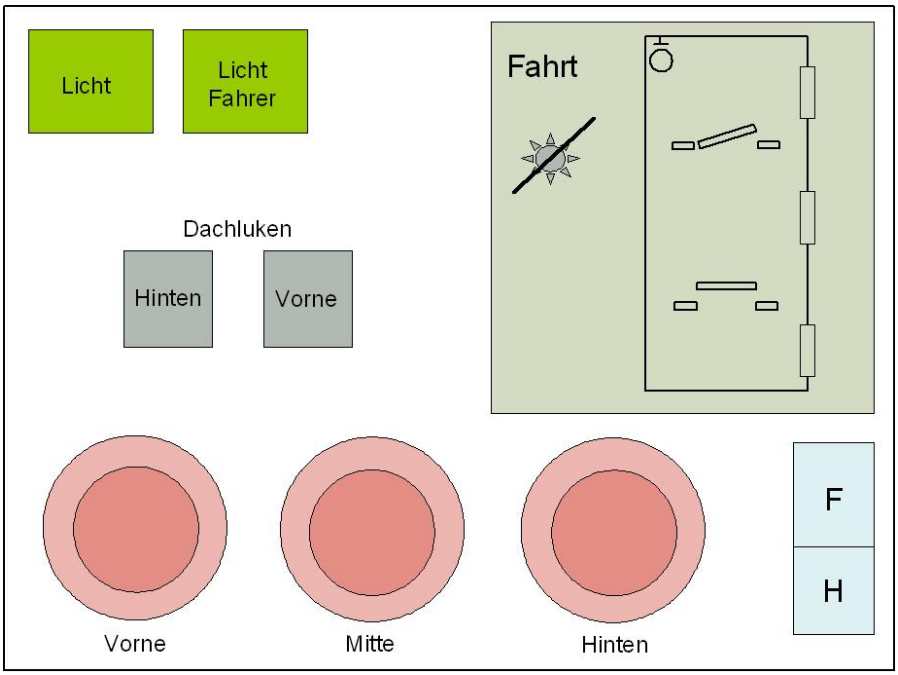


Abbildung 3.8.2: Übersicht über die Bedienkonsole Rechts

Mit der *Konsole Rechts* kann der Fahrer die Türen des Fahrzeugs bedienen. Hierzu befinden sich unten auf der Konsole drei Knöpfe, einer für jede Tür. Rechts daneben befindet sich der Schalter für den Haltestellenmodus. Das detaillierte Verhalten der Knöpfe und ihrer Aktuatoren wird in Abschnitt *3.1 Steuerung der Bustüren* erläutert.
Links oben befinden sich die Schalter für das Licht im Fahrzeuginnenraum, welche in Abschnitt *3.3 Steuerung des Fahrzeuginnenraumlichts* näher beschrieben werden.
Darunter sind die Knöpfe für die Steuerung der Dachluken angebracht. Sie werden in Abschnitt *3.4 Steuerung der Dachluken* detailliert betrachtet.
Rechts oben befindet sich ein LCD-Bildschirm, mit welchem der Fahrer über den Zustand der Türen, der Dachluken und des Innenlichts informiert wird. Auch etwaige Fehlermeldungen werden hier angezeigt.

3.9 Die Diagnoseeinheit

Mit Hilfe der Diagnoseeinheit kann ein Spezialist in einer Fachwerkstatt mit Hilfe eines Spezialgeräts Daten aus Steuergeräten auslesen. Das Gerät wird über eine spezielle Schnittstelle an den CAN-Bus des Fahrzeugs angeschlossen, der Benutzer hat dann die Möglichkeit, eine Diagnose aller Steuergeräte im Fahrzeug zu starten und die Ergebnisse auszulesen. Hierzu werden von den jeweiligen Steuergeräten bestimmte Daten über den CAN-Bus an das externe Diagnosegerät gesendet.
Die Diagnoseeinheit wird nur angeschlossen, wenn

- Die Zündung auf Stellung BETRIEB steht
- Die Geschwindigkeit des Fahrzeugs 0 km/h beträgt.

Folgende Daten müssen an die Diagnoseeinheit geschickt werden:

- der Kilometerstand des Fahrzeugs
- Ölstand
- Druck im Hydraulik- und Druckluftsystem (Bremsen)
- Daten des BSG:
    - Aufgetretene Fehler
- Fahrzeuginnenraumtemperatur und Außentemperatur
- Falls ein allgemeiner Fehler aufgetreten ist, so ist dieser und der Ort anzuzeigen

Mit Hilfe dieser Daten kann der Spezialist eine grobe Überprüfung der Funktionstüchtigkeit des Fahrzeugs durchführen.
Tritt während der Bedienung einer durch das BSG gesteuerten Komponente ein Fehler auf, so muss dieser im BSG gespeichert werden. Hierzu stehen 50 Speicherplätze zur Verfügung, in die die Fehler eingetragen werden. Mit der Diagnoseeinheit können diese Fehler ausgelesen werden.
Wird die Diagnoseeinheit nach erfolgter Diagnose ausgesteckt, so ist der gesamte Fehlerspeicher zu löschen.
Treten bis zur Diagnose mehr als 50 Fehler auf, so werden die ersten Fehler überschrieben, d.h. es findet eine zyklische Beschreibung der Speicherplätze statt.
Ein Eintrag auf einem Speicherplatz sieht hierbei wie folgt aus:

1. Byte: Code des Steuergeräts, für das BSG: 1
2. Byte: Teilbereich des Fehlers:
    a. Türen: 1
    b. Dachluken: 2
    c. Fahrersitz: 3
    d. Außenspiegel: 4
    e. Fenster: 5

3. Byte: Fehler:
    a. Blockade beim Aufmachen: 1
    b. Blockade beim Schließen: 2
    c. Wert außerhalb des gültigen Bereichs: 3

Die Fehlermeldung 152 bedeutet beispielsweise, dass das BSG einen Fehler beim Ansteuern des Fahrerfensters einen Fehler gemeldet hat: das Fenster wurde beim Schließen blockiert.

Kürzel Diagnoseeinheit	Beschreibung
DIAGNOSE_ANGE_C	Diese Bedingung ist wahr, wenn das externe Diagnosegerät an den CAN-Bus des Fahrzeugs angeschlossen ist.
DIAGNOSE_START_E	Über dieses Signal wird mitgeteilt, dass vom Benutzer eine Diagnose der Steuergeräte erwünscht ist und die entsprechenden Daten an das Diagnosegerät gesendet werden müssen.
DIAGNOSE_BUFFER_D	Dies ist ein Array mit 50 Plätzen à 3 Byte. In jedem Feld kann ein Fehlercode gespeichert werden. Dieser Code wird bei Bedarf an das Diagnosegerät gesendet.

Tabelle 3.9: Beschreibung der Signale für die Diagnoseeinheit

Produktwunsch 1

In diesem Dokument werden die Anforderungen für einen konkreten Bus beschrieben. Hierbei handelt es sich im Wesentlichen um ein Fahrzeug für den Einsatz im Liniendienst für den öffentlichen Personennahverkehr.
Grundsätzlich gilt für dieses Dokument: Alle Anforderungen, wie sie in der Kernfunktionalität beschrieben wurden, bleiben erhalten, sofern sie nicht ausdrücklich in diesem Dokument verändert bzw. erweitert werden.
Der Aufbau des Dokuments ist ähnlich dem der Kernfunktionalität. Allerdings werden nur noch Teile ausformuliert, in denen sich Änderungen ergeben haben.

Neu im Fahrzeug ist die Funktionalität des Benutzermanagements. Diese wird im Abschnitt *1.10 Das Benutzermanagement* oberflächlich und in Abschnitt *3.4 Das Benutzermanagement* ausführlich beschrieben.

Inhaltsverzeichnis

1. Überblick über das BSG

In diesem Abschnitt wird ein Überblick über die Funktionalität und die Eigenschaften des BSGs gegeben.

Die Kommunikation erfolgt genauso wie auch in der Kernfunktionalität beschrieben über den CAN-Bus. In Abbildung 1.1 ist jetzt auch das Benutzermanagement in der schematischen Übersicht des BSGs enthalten.

Abbildung 1.1: Schematische Übersicht über das BSG

1.1 Steuerung der Bustüren

Neben der Kernfunktionalität kommen hier weitere Anforderungen hinzu:
So müssen die beiden hinteren Türen im Fahrzeug nun im Haltestellenmodus auch von den Fahrgästen zu bedienen sein. Hierzu befinden sich an den Türen innen und außen jeweils Schalter, welche zu einem Öffnen der Türen führen. Das Schließen der Tür erfolgt automatisch bzw. durch den Fahrer.
Die Fahrertür besteht aus zwei Türflügeln, die in Zukunft separat vom Fahrer zu öffnen und schließen sind.
Außerdem wird im Fahrzeug ein Unfallsensor integriert. Dieser erkennt aus verschiedenen Parametern die Situation eines Unfalls und teilt dies dem BSG über den CAN-Bus mit. Daraufhin sind vom BSG bei Stillstand des Fahrzeugs sämtliche Türen automatisch zu öffnen.

Entsprechend dieser Anforderungen verändert sich natürlich auch die Konsole beim Fahrer. Hierzu sollte Abschnitt *3.3 Die Bedienkonsolen beim Fahrer* näher betrachtet werden.

Die genaue Beschreibung der Funktionalität der Türen ist im Abschnitt *3.1 Steuerung der Bustüren* dargelegt.

1.2 Steuerung der Klimaanlage

Die Klimaanlage funktioniert genauso wie in der Kernfunktionalität, hier sind keine Änderungen notwendig.

1.3 Steuerung des Fahrzeuginnenraumlichts

In diesem Abschnitt ergeben sich Änderungen an die Funktionalität.
Aus Komfortgründen für den Fahrer wird das Fahrgastinnenraumlicht hinten (also nicht beim Fahrer) an die Funktion des Abblendlichts gekoppelt. Schaltet der Fahrer das Fahrlicht des Fahrzeugs ein, so ist auch das Licht im Fahrgastraum vom BSG anzuschalten.
Der Fahrer kann über die Konsole die Funktion des Innenlichts überschreiben, das Licht also nach Belieben ein- bzw. ausschalten.
Die näheren Einzelheiten sind in Abschnitt *3.2 Steuerung des Fahrzeuginnenraumlichts* beschrieben.

1.4 Steuerung der Dachluken

In diesem Abschnitt ergeben sich keine Änderungen im Vergleich zur Kernfunktionalität.

1.5 Fahrersitzeinstellungen

Auch in diesem Abschnitt wird keine Veränderung gegenüber der Kernfunktionalität gemacht. Zu beachten ist allerdings, dass der Fahrersitz jetzt auch durch das im BSG befindliche Benutzermanagement angesteuert wird.

1.6 Einstellung der Außenspiegel

Für die Funktionalität der Außenspiegel siehe die Anforderungen in der Kernfunktionalität.

1.7 Bedienung des Fensters in der Fahrertür

Das Fenster ist in diesem Fahrzeug genauso zu bedienen wie in der Kernfunktionalität beschrieben.

1.8 Die Bedienkonsolen beim Fahrer

Entsprechend den neuen Anforderungen verändern sich auch die Bedienkonsolen (siehe auch Abbildung 1.8.1) des Fahrzeugs.

Mit der *Konsole Links* werden nun neben den beiden Außenspiegeln des Fahrzeugs und der Klimaanlage auch das Benutzermanagement gesteuert, mit der *Konsole Rechts* werden die Innenraumbeleuchtung, die Dachluken, die Türen und der Haltestellenmodus (samt Automatikmodus) eingestellt, wobei für die Steuerung der Türen Änderungen eingetreten sind.
Tritt in einer Funktion des BSG ein Fehler auf, so ist dies dem Fahrer im Display anzuzeigen Die ausführliche Beschreibung der Eigenschaften der Bedienkonsolen befinden sich in Abschnitt *3.3 Die Bedienkonsolen beim Fahrer*.

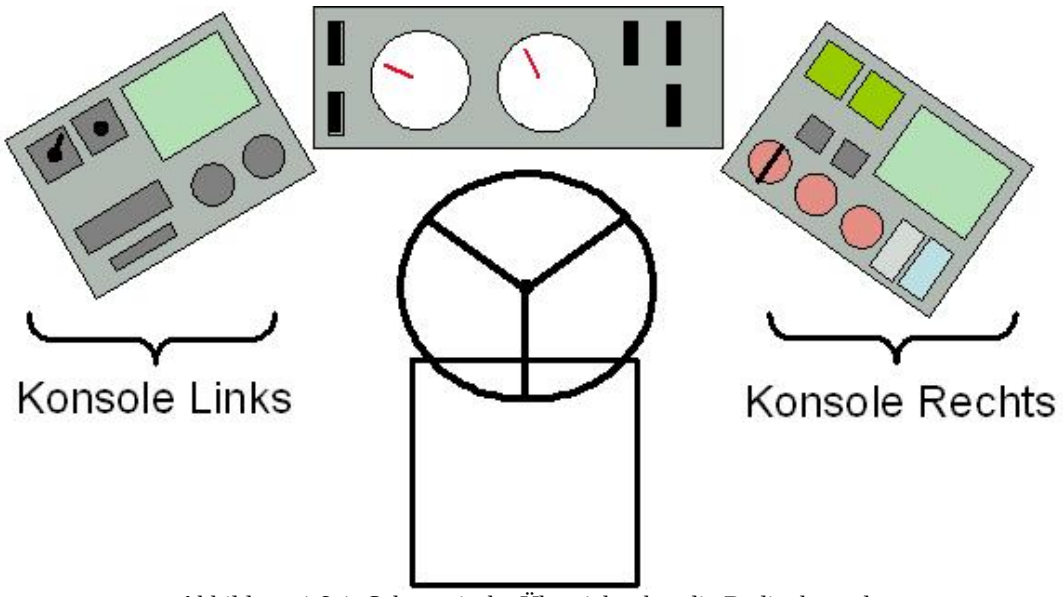


Abbildung 1.8.1: Schematische Übersicht über die Bedienkonsolen

1.9 Die Diagnoseeinheit

Für die Diagnoseeinheit ergeben sich keine Veränderungen im Vergleich zur Kernfunktionalität.

1.10 Das Benutzermanagement

Ohne ein Benutzermanagement muss jeder Fahrer den einstellbaren Fahrersitz vor Beginn jeder Fahrt überprüfen und gegebenenfalls neu einstellen. Diese Aufgaben nimmt das Benutzermanagement den Fahrern ab, in dem es diese Einstellungen speichert und bei Wiederbenutzung des Fahrzeugs durch den gleichen Fahrer dessen Einstellungen wiederherstellt.
Es stehen für vier Fahrer Speicherplätze zur Verfügung. Für die Einstellung gibt es vier Zifferntasten, die den vier benutzerdefinierten Einstellungen (Profile) entsprechen. Durch den Druck auf eine der Zifferntasten werden bei Stillstand des Fahrzeugs die entsprechenden Einstellungen wiederhergestellt.
Um ein Profil speichern zu können gibt es eine weitere Taste, die Taste Speichern. Die genaue Funktionalität des Benutzermanagements wird in Abschnitt *3.4 Das Benutzermanagement* beschrieben.

2. Hardware und Rahmenbedingungen

In diesem Abschnitt sind generelle Funktionen enthalten, die für die Funktionsweise des BSGs wichtig sind.
Die Funktionen für die Zündung, das Fahrzeuglicht und den Scheibenwischer können aus der Kernfunktionalität entnommen werden.
Zusätzlich wird im Fahrzeug ein Unfallsensor eingebaut, welcher einen Unfall erkennen kann und dieses an alle am CAN-Bus hängenden Geräten melden kann. Die Signale werden in Tabelle 2.1 beschrieben

Kürzel Unfallsensor	Beschreibung
UNFALL_ERKANNT_E	Dieses Event wird ausgelöst, wenn die Unfallerkennung einen Unfall erkannt hat. Dies erfolgt z.B. beim Auslösen eines Airbags oder wenn ein Drucksensor in einer Stoßstange extreme Werte meldet.

Tabelle 2.1 Signale des Unfallsensors

3. Ausführliche Beschreibung der geänderten Funktionalitäten

In diesem Abschnitt werden nur die vom BSG gesteuerten Komponenten mit geänderter Funktionalität ausführlich beschrieben.

3.1 Steuerung der Bustüren

Der Bus verfügt über drei Doppeltüren, welche über einen komplizierten Mechanismus geöffnet und geschlossen werden können. Hierzu stehen dem Fahrer drei Knöpfe zur Verfügung, mit denen er die Türen einzeln steuern kann (siehe Abbildung 3.1.1). Die Türen werden ab jetzt von vorne nach hinten durchnummeriert, d.h. die Fahrertür ist Nummer 1, die hintere Tür die Nummer 3.

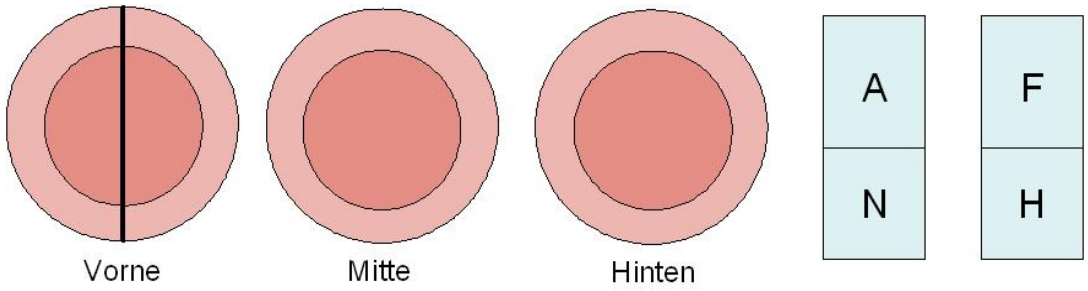


Abbildung 3.1.1: Bedienelement für den Haltestellenmodus und die Türen

Für die vordere Doppeltür gibt es nun einen zweigeteilten Schalter, welcher die beiden Türflügel symbolisiert, die nun einzeln zu steuern sind. Der zweigeteilte Taster funktioniert genauso wie die übrigen Taster, nur wird entsprechend nur ein Türflügel angesteuert. Dieser Taster hat nun entsprechend auch zwei Lampen im Inneren, die den Öffnungszustand der einzelnen Türflügel anzeigen.

Auf der rechten Seite in Abbildung 3.3.1 erkennt man nun einen weiteren Kippschalter mit der Beschriftung „A“ (Automatik) und „N“ (Normal). Mit diesem Schalter kann der Fahrer einstellen, ob Fahrgäste die beiden hinteren Türen auch mittels Knöpfen an der Innen- und Außenseite des Fahrzeugs selbstständig öffnen können (Automatikmodus). Dies funktioniert selbstverständlich nur im Haltestellenmodus (siehe Kernfunktionalität). Im Modus „Normal“ kann nur der Fahrer die Türen bedienen (hierfür siehe Beschreibung der Kernfunktionalität).
Im Automatikmodus kann der Kunde durch Knopfdruck signalisieren, dass die Fahrzeugtür geöffnet werden soll. Ist der Fahrzeug im Haltestellenmodus, so geht die entsprechende Tür sofort auf. Wenn sich das Fahrzeug im Fahrmodus befindet, so wird der Öffnungswunsch gespeichert und sobald der Haltestellenmodus aktiviert wird (und entsprechend der Schalter nach wie vor auf „A“ steht) die Tür geöffnet. In beiden Varianten schließt sich die Tür automatisch nach 3 Sekunden wieder, es sei denn es wird erneut ein Knopfdruck durch einen Fahrgast registriert. In diesem Fall wird der Zeitraum nach jedem Knopfdruck nach hinten verschoben. An jeder der beiden Doppeltüren im Fahrzeug (ausgenommen die Fahrertür) befindet sich Innen und Außen jeweils ein Druckknopf, mit dem Fahrgäste den Wunsch signalisieren können, dass die Tür geöffnet werden soll.
Der Fahrer kann die Türen schließen, in dem er den Fahrmodus einstellt (Achtung, Änderung zur Kernfunktionalität!) oder den Automatikbetrieb beendet. In beiden Fällen schließen sich

die Türen sofort. Zu beachten ist, dass selbstverständlich auch im Automatikbetrieb alle Sicherheitsbelange wie in der Kernfunktionalität beschrieben ihre volle Gültigkeit haben.
Die Lampen in den Türtastern müssen auch im Automatikmodus anzeigen, ob eine Tür offen oder geschlossen ist.

Durch den im Fahrzeug eingebauten Unfallsensor (siehe Abschnitt *2. Hardware und Rahmenbedingungen*) ist es möglich, Unfälle zu erkennen. Dies soll auch bei der Steuerung der Türen ausgenutzt werden: Nachdem ein Unfall erkannt worden ist, sind bei Stillstand des Fahrzeugs – egal in welchem Fahrmodus oder Betriebsmodus der Türen – alle Türen sofort zu öffnen.

Kürzel Türsteuerung	Beschreibung
HS_SWITCH_C	Gibt die Stellung des „Haltestellenkippschalters“ an. TRUE steht für „Haltestelle“, FALSE für „Fahrt“
FZG_TUER_TASTE_*x*_E	Das Event von der *Konsole Rechts*, welches beim Drücken auf die jeweilige Taste ausgelöst wird. *x* steht für 1L, 1R, 2 oder 3.
FZG_TUER_LICHT_*x*_C	Diese Bedingung ist wahr, wenn das Lämpchen im Türtaster an ist, mit ihr wird das Lämpchen gesteuert. *x* steht wiederum für 1R, 1L, 2 oder 3.
TUER_*x*_MOTOR_OPEN_E	Durch dieses Event wird der Motor der Tür *x* angehalten, die Tür *x* zu öffnen. Der Motor erkennt selbstständig, wann die Tür geöffnet ist und stoppt.
TUER_*x*_MOTOR_CLOSE_E	Analog zu oben weist dieses Signal den Motor der Tür *x* an, diese zu schließen. Wiederum erkennt der Motor, wann die Tür geschlossen ist und stoppt.
TUER_*x*_CLOSED_C	Ein Sensor reagiert, wann die Tür *x* komplett geschlossen ist. Dann wird die Bedingung für die Tür *x* auf wahr gesetzt. Sobald die Tür öffnet, wird die Bedingung auf falsch gesetzt. *x* steht wieder für 1R, 1L, 2 oder 3.
TUER_*x*_BLOCKED_ONCLOSE_C	Diese Bedingung wird wahr, wenn einer der Türflügel beim Schließen der Tür *x* auf einen Widerstand trifft. *x* für 1R, 1L, 2 oder 3.
TUER_*x*_BLOCKED_ONOPEN_C	Diese Bedingung wird wahr, wenn einer der Türflügel der Tür x beim Öffnen auf einen Widerstand trifft. *x* und ist ein Platzhalter analog wie oben.
AUTO_SWITCH_C	Gibt die Stellung des Automatikwahlschalters an. Ist wahr, wenn der Automatik-Modus an ist.
HALTE_BUTTON_*x*_E	Das Signal für den Türöffnungswunsch von innen. Achtung, *x* steht hierbei nur für 2 bzw. 3.
OPEN_BUTTON_*x*_E	Das Signal für einen Türöffnungswunsch von außen. *x* steht für 2 bzw. 3.

Tabelle 3.1: Beschreibung der Signale für die Türsteuerung

3.2 Steuerung des Fahrzeuginnenraumlichts

Der Innenraum des Fahrzeugs ist mit Lampen ausgestattet, um auch im Dunklen für genügend Sicherheit und Komfort im Fahrzeuginneren zu sorgen. Der Fahrer hat vorne bei sich am Tür-

raum ein Licht, um bei Nacht zahlenden Kunden genügend Licht zu bieten, das von den anderen Lampen getrennt zu bedienen ist.

Abbildung 3.2.1: Knöpfe für das Fahrzeuginnenlicht

An der *Konsole Rechts* befinden sich zwei Taster, mit denen der Fahrer das Innenraumlicht und das „Fahrerlicht“ getrennt voneinander bedienen kann (siehe auch Abbildung 3.2.1). Die Taster haben zwei Positionen, „gedrückt“ und „normal“, die abwechselnd durchgeschaltet werden.

Zu beachten ist, dass das Innenraumlicht in Zukunft an das Abblendlicht des Fahrzeugs gekoppelt ist. Sobald das Datum FAHRZEUG_LICHT_D ungleich 0 ist, so ist die Beleuchtung im Fahrzeuginnenraum (ausgenommen das Licht beim Fahrer, dieses ist weiterhin nur manuell zu steuern) anzuschalten. Hierbei ist nach folgenden Regeln zu verfahren:
Der Schalter „Licht“ (siehe Abbildung 3.2.1) wird durch das ans Fahrlicht gekoppelte Innenlicht nicht beeinflusst. Normalerweise muss der Busfahrer diesen Knopf nicht drücken, das Innenraumlicht wird praktisch automatisch durch das Fahrlicht gesteuert. Will der Fahrer das Licht trotz eingeschalteten Fahrlichts ausschalten, so muss er den Knopf drücken, das Licht im Innenraum geht dann aus. Entsprechend geht das Licht an, wenn das Abblendlicht dann ausgeschaltet wird. Durch ein erneutes Drücken wird das Innenraumlicht dann ausgeschaltet.

In Tabelle 3.2 werden die für die Steuerung des Fahrzeuginnenraumlichts benötigten Signale beschrieben.

Kürzel Fahrzeuglicht	Beschreibung
FZG_INNENLICHT_TASTE_*x*_C	Diese Bedingung ist wahr, wenn die jeweilige Taste im Modus „gedrückt“ ist. *x* steht für FAHRER bzw. REST.

Tabelle 3.2: Beschreibung der Signale für das Fahrzeuglicht

3.3 Die Bedienkonsolen beim Fahrer

Beim Fahrer sind links und rechts neben dem eigentlichen Armaturenbrett zwei Bedienkonsolen angebracht, die *Konsole Links* und die *Konsole Rechts*.
Abbildung 3.3.1 zeigt die *Konsole Links*.

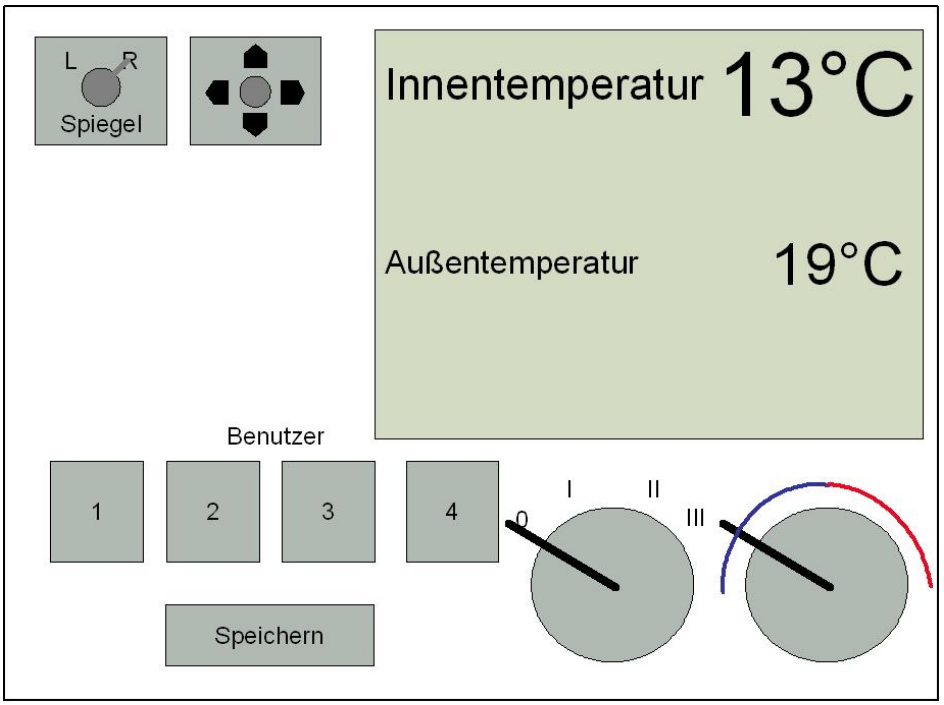


Abbildung 3.3.1: Übersicht über die Bedienkonsole Links

Neu in dieser Konsole sind die Bedienelemente für das Benutzermanagement. Mit den oberen Zifferntasten kann eine gespeicherte Sitzposition abgerufen werden, die untere Taste „Speichern“ dient zum Abspeichern der momentanen Sitzeinstellungen in einen Speicherplatz.

Abbildung 3.3.2 enthält die Übersicht über die *Konsole Rechts*.

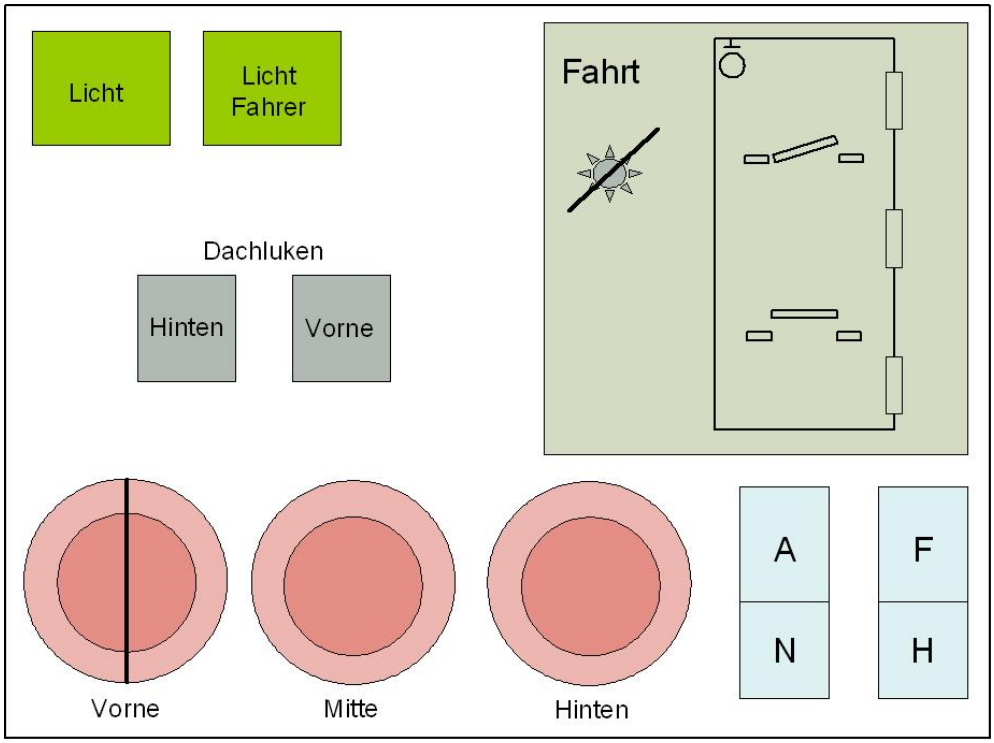


Abbildung 3.3.2: Übersicht über die Bedienkonsole Rechts

Mit der *Konsole Rechts* kann der Fahrer die Türen des Fahrzeugs bedienen. Der Taster für die Tür „Vorne“ ist jetzt in zwei Hälften unterteilt, welche die beiden Türflügel symbolisieren. Die beiden Hälften können unabhängig voneinander bedient werden. Rechts unten ist

ebenfalls ein neuer Taster für den Automatikmodus der Türen. In diesem Modus können die Fahrgäste die beiden hinteren Türen im Fahrzeug von selbst öffnen.

Das detaillierte Verhalten der Knöpfe und ihrer Aktuatoren wird in Abschnitt *3.1 Steuerung der Bustüren* erläutert.

3.4 Das Benutzermanagement

Mit dem Benutzermanagement können bis zu vier verschiedene Fahrer ihre Einstellungen im Fahrzeug speichern und einfach per Knopfdruck abrufen. Das BSG sorgt dann dafür, dass die entsprechenden Positionen des Sitzes wiederhergestellt werden.
Das BSG reagiert auf Benutzerwünsche nur, wenn

- die Zündung auf Stellung BETRIEB steht
- die Geschwindigkeit des Fahrzeugs weniger als 10 km/h beträgt

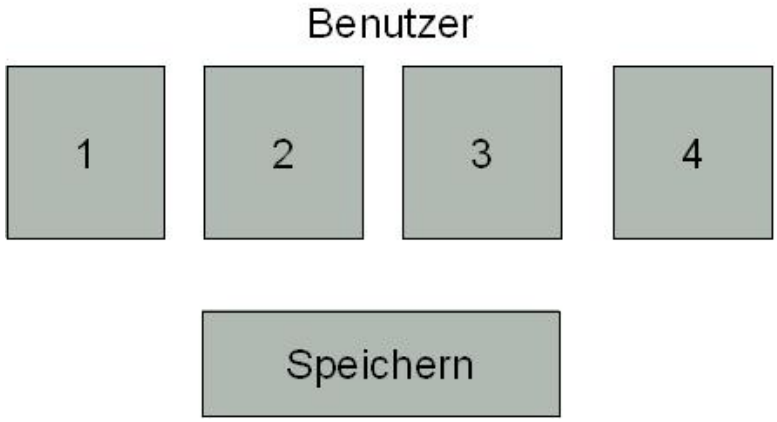


Abbildung 3.4.1: Bedienfelder für das Benutzermanagement

In Abbildung 3.4.1 ist das Bedienfeld für das Benutzermanagementsystem dargestellt. Mit den Zifferntasten 1 bis 4 können vier eingespeicherte Konfigurationen abgerufen werden (im Folgenden auch Speicherplätze genannt). Mit der Taste „Speichern“ kann eine Konfiguration einem bestimmten Speicherplatz zugewiesen werden.
Um einen Speicherplatz zu belegen, muss die Taste „Speichern“ gedrückt gehalten werden und anschließend innerhalb von 2 Sekunden eine der Zifferntasten. Die momentan eingestellten Werte für den Fahrersitz werden dann unter diesem Speicherplatz abgelegt. Wird innerhalb dieser 2 Sekunden keine Zifferntaste gedrückt, so muss die Speichern-Taste erst wieder losgelassen werden, um einen erneuten Speichervorgang zu starten. Während die Speichern-Taste gedrückt ist, werden keine Signale der manuellen Sitzverstellung akzeptiert.
Um eine Speicherung abzurufen, muss eine der Zifferntasten für mindestens eine halbe Sekunde gedrückt werden. Wir die Taste danach losgelassen, so fährt das BSG den Sitz in die abgespeicherten Positionen. Dabei wird folgende Reihenfolge eingehalten:

- Parallel wird der Sitz um Zeit zu sparen in die horizontal und vertikal richtige Position gefahren.
- Hat der Sitz die richtige Position erreicht, so wird als letzter Schritt die Lehne in die richtige Stellung gebracht.

Das Benutzermanagement akzeptiert weitere Befehle durch den Fahrer erst dann wieder, wenn der Einstellungsvorgang abgeschlossen ist. Bis dahin werden alle Tastendrücke ignoriert, auch die Taster für die manuelle Sitzverstellung werden erst wieder berücksichtigt, wenn der Sitz in die entsprechende Position angefahren hat. Selbstverständlich gelten für die

Fahrersitzeinstellungen die gleichen Vorgaben für das BSG wie für andere Benutzer, insbesondere sind alle Sicherheitsaspekte zu berücksichtigen (siehe Kernfunktionalität).
Das manuelle Einstellen des Sitzes durch den Fahrer mit Hilfe der entsprechenden Bedienelemente wird sofort abgebrochen, wenn das Benutzermanagement eine gespeicherte Konfiguration einstellt.
Das Einstellen der Positionen durch das BSG im Benutzermanagementbetrieb wird abgebrochen, falls eine der Positionen nicht innerhalb von 30 Sekunden erreicht werden kann. In diesem Fall wird im Display der Konsole Links die Meldung „Benutzermanagement Fehler" eingeblendet. Der Fahrer kann jederzeit wieder eine Speicherposition abrufen. Gelingt das Einstellen der Positionen, so muss die Meldung im Display verschwinden.
Tabelle 3.4 enthält die für das Benutzermanagement benötigten Signale.

Kürzel Benutzermanagement	Beschreibung
BENUTZ_MGT_POS_*x*_C	Diese Bedingung ist wahr, wenn eine Zifferntaste auf dem Bedienfeld gedrückt wird. *x* steht hierbei für die Ziffern 1 bis 4.
BENUTZ_MGT_SAFE_C	Diese Bedingung ist wahr, wenn die Taste „Speichern" des Bedienfelds gedrückt wird.
DISPLAY_ERROR_D	String mit einer Länge von 20 Zeichen zur Fehlerausgabe auf dem Display der *Konsole Links*.

Tabelle 3.4: Beschreibung der Signale für das Benutzermanagement

Produktwunsch 2

In diesem Dokument werden die Anforderungen für einen konkreten Bus beschrieben. Hierbei handelt es sich im Wesentlichen um ein Fahrzeug für den Einsatz im Reisebusverkehr.
Grundsätzlich gilt für dieses Dokument: Alle Anforderungen, wie sie in der Kernfunktionalität beschrieben wurden, bleiben erhalten, sofern sie nicht ausdrücklich in diesem Dokument verändert bzw. erweitert werden.
Der Aufbau des Dokuments ist ähnlich dem der Kernfunktionalität. Allerdings werden nur noch Teile ausformuliert, in denen sich Änderungen ergeben haben.

Neu im Fahrzeug ist die Funktionalität des Benutzermanagements. Diese wird oberflächlich in Abschnitt *1.10 Das Benutzermanagement* und ausführlich in Abschnitt *3.6 Das Benutzermanagement* beschrieben.

Inhaltsverzeichnis

1. Überblick über das BSG

In diesem Abschnitt wird ein Überblick über die Funktionalität und die Eigenschaften des BSG gegeben.

Die Kommunikation erfolgt genauso wie auch in der Kernfunktionalität beschrieben über den CAN-Bus. In Abbildung 1.1 ist jetzt auch das Benutzermanagement in der schematischen Übersicht des BSGs enthalten.

Abbildung 1.1: Schematische Übersicht über das BSG

1.1 Steuerung der Bustüren

In diesem Abschnitt ergeben sich keine Änderungen im Vergleich zur Kernfunktionalität.

1.2 Steuerung der Klimaanlage

In diesem Fahrzeug ist nun eine echte Klimaautomatik zu realisieren, d.h. der Fahrer kann eine gewünschte Innenraumtemperatur einstellen, die Automatik regelt dann die Temperatur.
Für die genaue Beschreibung der Klimaanlage bitte unter Abschnitt *3.1 Steuerung der Klimaanlage* auf Seite 6 nachsehen.

1.3 Steuerung des Fahrzeuginnenraumlichts

In diesem Abschnitt ergeben sich keine Änderungen im Vergleich zur Kernfunktionalität.

1.4 Steuerung der Dachluken

Die Dachluken sind in diesem Fahrzeug an die Funktion des Scheibenwischers zu koppeln. Ist der Scheibenwischer mit der Geschwindigkeit „Normal“ an, so können die Dachluken nur noch hinten (in Fahrtrichtung) geöffnet werden, da sonst zuviel Regenwasser ins Fahrzeug dringen könnte. Ist der Scheibenwischer im Modus „Schnell“, so lassen sich die Dachluken nicht mehr öffnen.
Die detaillierte Beschreibung der Funktionalität der Dachluken befindet sich in Abschnitt *3.2 Steuerung der Dachluken*.

1.5 Fahrersitzeinstellungen

In diesem Abschnitt ergeben sich keine Änderungen im Vergleich zur Kernfunktionalität.

1.6 Einstellung der Außenspiegel

In die Außenspiegel des Fahrzeugs werden hinter die Spiegelflächen jetzt Heizmatten integriert, d.h. die Außenspiegel sind – in Abhängigkeit von der Außentemperatur – beheizbar. Zu beachten ist auch, dass die Außenspiegel jetzt auch durch das im BSG befindliche Benutzermanagement angesteuert werden.
Eine detaillierte Beschreibung der erweiterten Funktionalität der Außenspiegel befindet sich in Abschnitt *3.3 Einstellung der Außenspiegel*.

1.7 Bedienung des Fensters in der Fahrertür

Die Kernfunktionalität wird hier um eine weitere Funktion erweitert. So muss nun eine Komfortfunktion für die Fensterheberbedienung eingebaut werden.
Nähere Einzelheiten werden in Abschnitt *3.4 Bedienung des Fensters in der Fahrertür* erläutert.

1.8 Die Bedienkonsolen beim Fahrer

Entsprechend den neuen Anforderungen verändern sich auch die Bedienkonsolen (siehe auch Abbildung 1.8.1) des Fahrzeugs.
Mit der *Konsole Links* werden die beiden Außenspiegel des Fahrzeugs, das Benutzermanagement sowie die Klimaanlage gesteuert, mit der *Konsole Rechts* werden die Innenraumbeleuchtung, die Dachluken, die Türen und der Haltestellenmodus eingestellt.
Das Fenster in der Fahrertür wird über einen Schalter, welcher in der Tür eingelassen ist, gesteuert, der Fahrersitz über Taster am Sitz.
Die ausführliche Beschreibung der Eigenschaften der Bedienkonsolen befinden sich in Abschnitt *3.5 Die Bedienkonsolen beim Fahrer*.

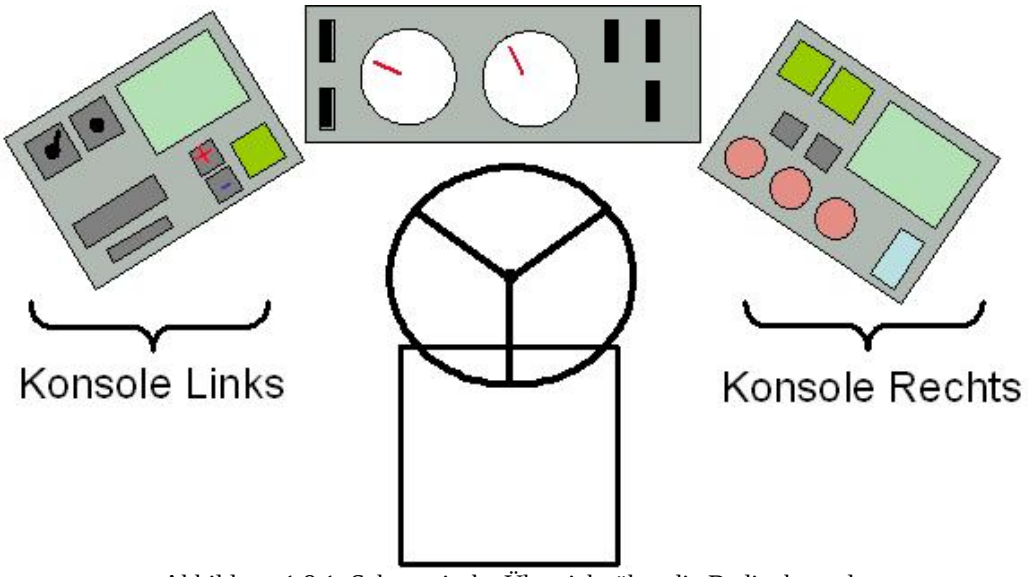


Abbildung 1.8.1: Schematische Übersicht über die Bedienkonsolen

1.9 Die Diagnoseeinheit

Die Funktionalität der Diagnoseeinheit ist die gleiche wie im Dokument für die Kernfunktionalität beschrieben.

1.10 Das Benutzermanagement

Ohne ein Benutzermanagement muss jeder Fahrer die einstellbaren Außenspiegel vor Beginn jeder Fahrt überprüfen und gegebenenfalls neu einstellen. Diese Aufgaben nimmt das Benutzermanagement den Fahrern ab, in dem es diese Einstellungen speichert und bei Wiederbenutzung des Fahrzeugs durch den gleichen Fahrer dessen Einstellungen wiederherstellt.
Es stehen für vier Fahrer Speicherplätze zur Verfügung. Für die Einstellung gibt es vier Zifferntasten, die den vier benutzerdefinierten Einstellungen (Profile) entsprechen. Durch den Druck auf eine der Zifferntasten werden bei Stillstand des Fahrzeugs die entsprechenden Einstellungen wiederhergestellt.
Um ein Profil speichern zu können gibt es eine weitere Taste, die Taste Speichern. Die genaue Funktionalität des Benutzermanagements wird in Abschnitt *3.6 Das Benutzermanagement* beschrieben.

2. Hardware und Rahmenbedingungen

In diesem Abschnitt sind generelle Funktionen enthalten, die für die Funktionsweise des BSGs wichtig sind.
Die Funktionen für die Zündung, das Fahrzeuglicht und den Scheibenwischer können aus der Kernfunktionalität entnommen werden.

3. Ausführliche Beschreibung der Funktionalitäten

In diesem Abschnitt werden nur die vom BSG gesteuerten Komponenten mit geänderter Funktionalität ausführlich beschrieben.

3.1 Steuerung der Klimaanlage

In diesem Fahrzeug ist eine Klimaautomatik eingebaut. Entsprechend verändert sich auch die *Konsole Links*. Für die Bedienung stehen dem Fahrer die in Abbildung 3.1.1 dargestellten Knöpfe zur Verfügung.

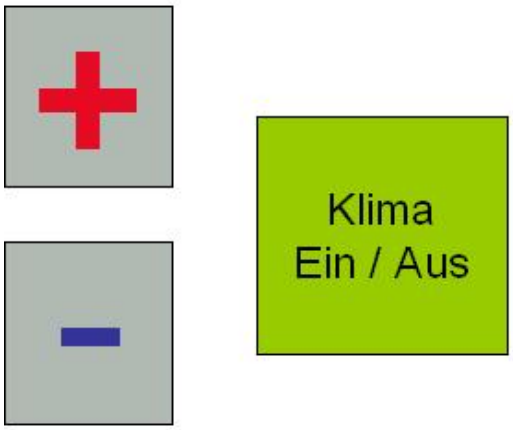


Abbildung 3.1.1: Bedienelement für die Klimaanlage.

Mit dem Taster auf der rechten Seite kann der Fahrer die Klimaautomatik ein- bzw. ausschalten. Der Taster hat zwei Positionen, „gedrückt“ und „normal“, die abwechselnd durchgeschaltet werden. In der Position „gedrückt“ ist die Klimaanlage eingeschaltet.
Mit den Knöpfen auf der linken Seite kann der Fahrer einen Innentemperaturwunsch einstellen. Durch einmaliges Drücken dieser Tasten wird die Solltemperatur um 0,1 erhöht bzw. erniedrigt. Werden beide Tasten gleichzeitig gedrückt, so verändert sich die Solltemperatur nicht. Beim Start des Fahrzeugs beträgt die Wunschtemperatur 19,0 Grad Celsius. Wenn der Fahrer länger als 1 Sekunde auf der Plus- bzw. Minus-Taste bleibt, dann erhöht bzw. vermindert sich die Solltemperatur nach dieser Sekunde um 0,1 pro halbe Sekunde. Damit ist eine schnelle Änderung der gewünschten Innentemperatur möglich.
Die Temperaturregelung wird durch das BSG vorgenommen: Die Automatik heizt, wenn die tatsächliche Innentemperatur kleiner als die Wunschtemperatur ist und kühlt, wenn die Innentemperatur größer als die Solltemperatur ist.
Für das Heizen bzw. Kühlen stehen drei Gebläsestufen zur Verfügung:

- Stufe 0 (Aus): Die Differenz zwischen Ist- und Solltemperatur <= 0,2
- Stufe 1: Die Differenz zwischen Ist- und Solltemperatur > 0,2
- Stufe 2: Die Differenz zwischen Ist- und Solltemperatur > 1,5
- Stufe 3: Die Differenz zwischen Ist- und Solltemperatur > 2,5

Die folgende Tabelle 3.1 enthält alle Signale, die für die neue Klimaautomatik im Fahrzeug benötigt werden.

Kürzel Klimaautomatik	Beschreibung
FZG_KLIMA_TASTE_EIN_C	Die Bedingung ist wahr, wenn der Taster in der Stellung „gedrückt“ ist, die Klimaautomatik also an sein soll.
WUNSCH_INNEN_TMP_D	Dieses Datum enthält eine Real-Variable mit der eingestellten Wunschtemperatur. Sie muss bei Tastendruck angepasst werden. Diese Variable benutzt auch das Display, um die Solltemperatur anzuzeigen.
WUNSCH_INNEN_UP_C	Die Bedingung ist wahr, wenn der Taster „+“ gedrückt wird.
WUNSCH_INNEN_DOWN_C	Diese Bedingung ist wahr, wenn die Minus-Taste gedrückt wird.
BURN(*Stufe*)	Betreibt das Gebläse mit Heizung mit der entsprechenden *Stufe*.
COOL(*Stufe*)	Betreibt das Gebläse und kühlt entsprechend den Innenraum mit der Intensität *Stufe* (0 bis 3).

Tabelle 3.1: Beschreibung der Signale für die Klimaanlage

Das Signal für die Innentemperatur kann in Abschnitt 2 der Beschreibung der Kernfunktionalität nachgeschlagen werden.

3.2 Steuerung der Dachluken

In diesem Abschnitt wird die Steuerung der Dachluken in dem Bus von einer Bedienkonsole beim Fahrer aus beschrieben.
Die Dachluken dürfen nur angesteuert werden, wenn die Zündung auf Stellung BETRIEB steht.
Im Fahrzeug sind zwei Dachluken eingebaut. Diese dienen im Notfall bei beispielsweise einem Umfallen des Fahrzeugs auf die Seite auch als Notausgänge, indem die Passagiere die Luken komplett vom Fahrzeugdach abtrennen und durch das Loch ins Freie klettern können. Die Luken haben vorne und hinten in Fahrtrichtung jeweils einen Elektromotor, so dass sie flexibel zu öffnen sind (siehe Abbildung 3.2.1).

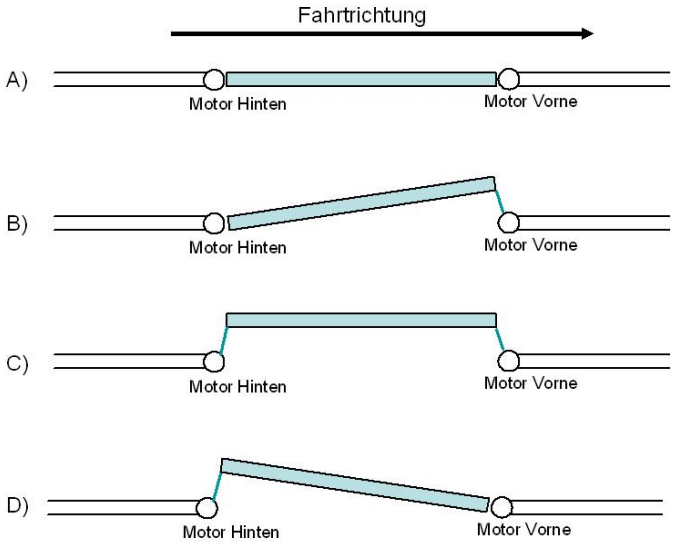


Abbildung 3.2.1: Schematischer Querschnitt durch das Fahrzeugdach quer in Fahrtrichtung

In früheren Busmodellen konnten diese Dachluken von Hand (und damit von Fahrgästen) bedient werden. In Zukunft soll die Ansteuerung dieser Luken durch das Bustürsteuergerät (BSG) erfolgen.
Der Fahrer kann über die Bedienkonsole folgende Modi einstellen:

- Geschlossen: Die Dachluke ist geschlossen (siehe Abbildung 3.2.1 A).
- Vorne Offen: Die Luke wird nur vorne geöffnet. Hierzu wird der vordere Elektromotor angesteuert. Dies führt zu einem starken Luftzug in das Fahrzeug (siehe Abbildung 3.2.1 B).
- Beide Offen: Die Luke wird vorne und hinten geöffnet, sie „schwebt“ praktisch parallel zum Fahrzeugdach (siehe Abbildung 3.2.1 C).
- Hinten Offen: Die Luke wird nur hinten geöffnet (siehe Abbildung 3.2.1 D).

Aufbau des Mechanismus

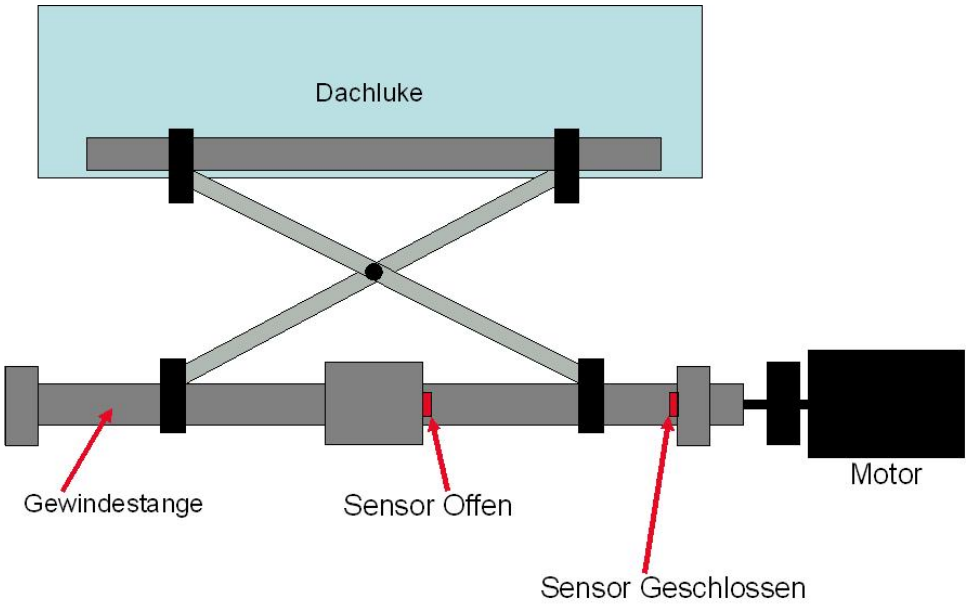


Abbildung 3.2.3 : Schließ- und Öffnungsmechanismus der Dachluken (Schematisch)

Ein Motor treibt eine Gewindestange an, auf der zwei Muttern laufen. Durch Drehung der Gewindestange werden die Muttern entweder vom Mittelpunkt weg geschoben, dann schließt sich die Luke, oder zum Mittelpunkt hingeschoben, entsprechend öffnet sich die Dachluke.
Zwei Sensoren (*Sensor Offen*, *Sensor Geschlossen*) zeigen an, wann die Luke auf dieser Seite vollständig geschlossen bzw. offen ist.
Zum Öffnen der Dachluke in diesem Bereich wird der Motor in die entsprechende Richtung gestartet, sobald der Sensor *Sensor Offen* anspricht wird der Motor gestoppt, die Dachluke ist offen. Umgekehrtes gilt für das Schließen.
Wird der Motor gestartet, aber nach drei Sekunden hat der entsprechende Sensor noch nicht reagiert, so ist von einem Fehler auszugehen.

An der Bedienkonsole vorne beim Fahrer befinden sich zwei Knöpfe für die Bedienung der Dachluken und ein Display, in welchem unter anderem der Öffnungszustand der Dachluken angezeigt wird (Abbildung 3.2.2).

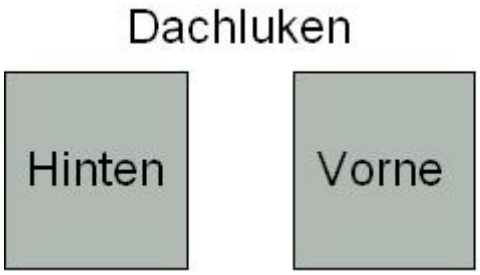


Abbildung 3.2.2: Bedienelement für die Steuerung der Dachluken

Mit den beiden Knöpfen wird der „Öffnungszustand“ für die beiden Dachluken zyklisch durchgeschaltet, und zwar in dieser Reihenfolge:

- Geschlossen (siehe Abbildung 3.2.1 A)
- Vorne offen (Abbildung 3.2.1 B)
- Beide offen (Abbildung 3.2.1 C)
- Hinten offen (Abbildung 3.2.1 D)

Mit jedem Knopfdruck des Fahrers wird die entsprechende Dachluke in die Position gefahren, die zyklisch an der Reihe ist. Ist also bspw. die vordere Luke geschlossen und der Fahrer drückt einmal auf den Knopf, so fährt die Luke in die Stellung „Vorne offen“.
Bei der Bedienung durch den Fahrer sind folgende Funktionalitäten zu berücksichtigen:

- Im Display muss nach jedem Tastendruck durch den Fahrer sofort die neue Stellung der jeweiligen Dachluke angezeigt werden.
- Die Luke wird allerdings erst in diese Position gefahren, wenn der Fahrer mindestens eine Sekunde die Taste nicht mehr gedrückt hat. So wird vermieden, dass durch dreimaliges Drücken der Taste kurz hintereinander die Luke immer wieder neu angesteuert werden muss. Der Fahrer hat dadurch auch die Möglichkeit Schritte bei der Reihenfolge zu „überspringen“ (Beispiel: die Luke ist auf „Beide Offen“ und soll jetzt ganz geschlossen werden: der Fahrer drückt zweimal kurz die Taste, nach einer Sekunde wird die Luke in die zuletzt eingestellte Position gefahren).

Selbstverständlich sind gewisse Sicherheitsstandards einzuhalten. So muss bei Blockierung der Bewegung in eine bestimmte Richtung die Bewegung sofort gestoppt werden. Dies wird vom Motor übernommen, der bei zu großer Belastung sofort selbstständig stoppt und ein entsprechendes Signal an das Steuergerät sendet. Von einem Fehler ist auch auszugehen, wenn die Dachluke innerhalb von 3 Sekunden nicht auf einen Befehl des Fahrers reagiert, d.h. z.B. die Luke hinten auffahren soll, der entsprechende Sensor aber nach spätestens drei Sekunden nicht anspricht.
In der folgenden Tabelle werden alle Events, Bedingungen, Sensorsignale und Motoransteuerungen für das Steuergerät mit zwei Dachluken (die vordere wird ab sofort mit Luke 1, die hintere mit Luke 2 bezeichnet) spezifiziert.

Kürzel Motorsteuerung	Beschreibung
MOTOR_DL_*x*_*y*_*z*_E	Dies ist das Event für die Motorsteuerung an einer Dachluke. *x* steht hierbei für 1 oder 2 (Dachluke 1 oder 2), *y* für V (Vorne) oder H (Hinten) und *z* für ZU oder AUF. Das Signal **MOTOR_DL_1_V_AUF_E** bedeutet also, dass an den vorderen Motor der Dachluke 1 das Signal zum Aufmachen geschickt wird.
MOTOR_*x*_*y*_STOPPED_E	Der Motor schickt ein Signal, wenn er stoppt. Dies passiert z.B. wenn die Endposition an der Gewindewelle erreicht wird oder der Motor blockiert wird. *x* steht für 1 oder 2

Kürzel Motorsteuerung	Beschreibung
	(Luke 1 oder 2), *y* für V (Vorne) oder H (Hinten).
SENSOR_DL_*x*_*y*_*z*_C	Eine Bedingung für die Sensoren an den Dachluken. Sie wird wahr, wenn am Sensor ein Signal anliegt. *x* steht hierbei für 1 oder 2 (Dachluke 1 oder 2), *y* für V (Vorne) oder H (Hinten) und *z* für GESCHLOSSEN oder OFFEN. Wenn also **SENSOR_DL_1_V_OFFEN_C** wahr ist, so ist die Dachluke 1 Vorne offen.
Kürzel Display-Steuerung	Beschreibung
DL_*x*_BUTTON_E	Das Event für die Knöpfe an der Bedienkonsole zur Steuerung der Dachluken (zyklisch). *x* steht für 1 oder 2 (Luke 1 und Luke 2). **DL_1_BUTTON_EVENT** bedeutet, dass der Knopf für die Luke 1 gedrückt wurde.
DL_*x*_STATUS_D	Hierüber wird an das Display der Status der einzelnen Dachluken ausgegeben, wobei *x* für 1 oder 2 (Dachluke 1 oder 2) steht. Folgende Werte kann der Status haben: - 0: Dachluke geschlossen - 1: Vorne Offen - 2: Beide Offen - 3: Hinten Offen
DL_*x*_ERROR_C	Diese Bedingung wird wahr, wenn an der Dachluke *x* ein Fehler festgestellt wurde.

Tabelle 3.2: Beschreibung der Signale für die Dachluken

Die Dachluken müssen in diesem Fahrzeug an den Scheibenwischer gekoppelt werden. Die Signale des Scheibenwischers können aus Abschnitt 2 der Kernfunktionalität abgelesen werden.
Ist der Scheibenwischer im Arbeitsmodus „Normal an“, so dürfen die Dachluken nur noch hinten (in Fahrtrichtung gesehen) aufgemacht werden, da sonst Wasser in den Fahrgastraum gelangen könnte (siehe Abbildung 3.2.1 D). Sind die Dachluken geöffnet und der Scheibenwischer wird in diesem Modus betrieben, so sind die Dachluken in die Position „hinten offen“ zu fahren und das Display muss aktualisiert werden. Der Fahrer kann mit seinen Bedienknöpfen nur noch zwischen den Stellungen „geschlossen“ und „hinten offen“ wählen.
Ist die Betriebstufe des Scheibenwischers „Schnell an“, so sind die Dachluken im geöffneten Fall sofort zu schließen, sie können auch durch den Fahrer nicht wieder geöffnet werden. Das Display ist entsprechend zu aktualisieren. Erst nach dem der Scheibenwischer in einem anderen Modus bedient wird, kann der Busfahrer die Dachluken wieder steuern.

3.3 Einstellung der Außenspiegel

Mit dem BSG kann der Fahrer die Position der beiden Außenspiegel im Fahrzeug einstellen. Die Außenspiegel im Fahrzeug dürfen nur eingestellt werden (d.h. die Spiegelmotoren dürfen angesteuert werden), wenn

- die Zündung auf Stellung BETRIEB steht
- die Sitzverstellung nicht betätigt wird

Jeder der beiden Spiegel (zur Unterscheidung im Folgenden *Spiegel_Links* und *Spiegel_Rechts* in Fahrtrichtung genannt) kann in horizontaler und vertikaler Richtung eingestellt werden. Im Spiegelgehäuse befinden sich Sensoren, welche die momentane Stellung der Spiegelscheibe erfassen. Da keinerlei Endsensoren angebracht sind, dürfen die

Werte der Spiegelsensoren einen bestimmten Wertebereich determiniert durch das Gehäuse nicht verlassen. Die Kontrolle ist Aufgabe des BSGs.
Für die Ansteuerung stehen dem Fahrer auf der *Konsole Links* die zwei in Abbildung 3.3.1 gezeigten Elemente zur Verfügung.

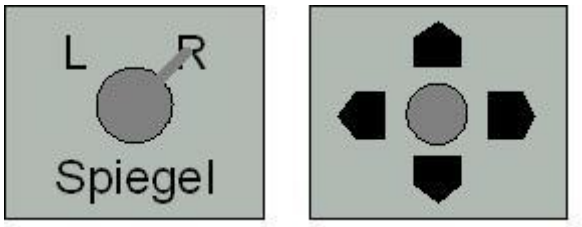


Abbildung 3.3.1: Bedienfelder für die Spiegeleinstellung

Mit dem Element auf der linken Seite wird eingestellt, ob der *Spiegel_Links* (Stellung auf L) oder der *Spiegel_Rechts* (Stellung auf R) angesteuert wird. Im rechten Element kann der Fahrer mittels einer Art kleiner Joystick den Spiegel in die gewünschte Richtung bewegen. Wird der Joystick zum Beispiel nach rechts gedrückt, so dreht sich die entsprechende Spiegelscheibe um die vertikale Achse nach rechts. Wird der kleine Joystick losgelassen, so kehrt er automatisch in die Ruhestellung in die Mitte zurück. Der Schalter erkennt nur eine Bewegungsrichtung. Der jeweilige Spiegel fährt nur, solange entweder der Joystick in die Richtung gedrückt wird oder der gültige Wertebereich nicht verlassen wird.
Der Wertebereich für die beiden Spiegel ist jeweils der Gleiche:

- Horizontal: -20° (Anschlag ganz links) bis +20° (Anschlag ganz rechts)
- Vertikal: -15° (Anschlag ganz unten) bis +15° (Anschlag ganz oben)

Pro Halbesekunde wird der Spiegel durch den Motor um +/-2° in vertikaler oder horizontaler Richtung bewegt.

Kürzel Außenspiegel	Beschreibung
SPIEGEL_ANSTEUERN_LINKS_C	Eine Bedingung, welcher Spiegel momentan eingestellt werden kann. Ist wahr, wenn Spiegel_Links angesteuert wird, falsch wenn der rechte Außenspiegel angesprochen wird (siehe Abbildung 3.7.1, linkes Element).
SPIEGEL_*y*_TASTER_C	Diese Bedingungen werden durch den kleinen Joystick im Bedienfeld auf der *Konsole Links* durch drücken des Joysticks in die jeweilige Richtung gesetzt. y steht hierbei für: - HOCH: Joystick wird nach oben gedrückt - RUNTER: Joystick wird nach unten gedrückt - RECHTS: Joystick wird nach rechts gedrückt - LINKS: Joystick wird nach links gedrückt
SPIEGEL_*x*_MOTOR_*y*_E	Die Signale für die Steuerung der Spiegelstellmotoren. *x* steht für LINKS und RECHTS, *y* kann folgende Werte annehmen: - HOCH: Motor dreht Spiegel nach oben - RUNTER: Motor dreht Spiegel nach unten - RECHTS: Motor dreht Spiegel nach rechts - LINKS: Motor dreht Spiegel nach links.
SPIEGEL_*x*_MOTOR_STOP_E	Signal, um den entsprechenden Spiegelmotor zu stoppen. *x* steht hierbei für LINKS bzw. RECHTS
SPIEGEL_*x*_*y*_POS_D	Dieses Datum enthält die aktuelle Position des

Kürzel Außenspiegel	Beschreibung
	Spiegels *x* in vertikaler bzw. horizontaler Richtung. *x* steht für LINKS bzw. RECHTS, *y* für HOR bzw. VERT.

Tabelle 3.3.1: Beschreibung der Signale für die Außenspiegel

Zu beachten ist, dass das BSG für die Positionen der Spiegel entsprechend Speicherstellen zur Verfügung stellt und die Werte fortlaufend überwacht. Außerdem kann das Benutzermanagement die Spiegel steuern, in diesem Fall sind gewisse Einschränkungen zu berücksichtigen (siehe Abschnitt *3.6 Das Benutzermanagement*).

Fällt die Temperatur außerhalb des Fahrzeugs auf unter 3 Grad Celsius, so muss automatisch die Spiegelheizung aktiviert werden. Bei einer Temperatur von 4 Grad Celsius wird die Spiegelheizung wieder deaktiviert. In beiden Spiegeln sind hinter der Scheibe Heizmatten eingebaut, welche durch Erwärmung ein Einfrieren der Spiegel verhindern. In Tabelle 3.3.2 werden die Signale erläutert, welche für die Spiegelheizung notwendig sind.

Kürzel Außenspiegel - Heizung	Beschreibung
SPIEGEL_*x*_HEAT_C	Eine Bedingung für die Heizmatte im jeweiligen Spiegel. Ist die Bedingung wahr, so wird der Spiegel beheizt. *x* steht für LINKS und RECHTS.

Tabelle 3.3.2: Beschreibung der Signale für die Außenspiegel-Heizung

3.4 Bedienung des Fensters in der Fahrertür

Aus Komfortgründen wird nun eine weitere Funktion in den Fensterheber integriert: Wird eine Taste des Fensterhebers länger als 0.5 Sekunden in eine bestimmte Richtung gedrückt, so kann die Taste auch losgelassen werden, die Scheibe fährt dann vollständig nach oben bzw. unten. Die Bewegung wird gestoppt wenn die Scheibe ganz oben bzw. unten ist oder wenn der Fenstertaster in die Gegenrichtung betätigt wird.
Eine Kontrolle von Fehlern findet bis auf die Widerstandskontrolle nicht statt.

Kürzel Fensterheber	Beschreibung
FENSTER_MOTOR_*x*_E	Ein Event für den Motor des Fensterhebers. *x* steht hierbei für AUF oder ZU, je nachdem in welche Richtung (oben bzw. unten) die Scheibe gefahren werden soll. Durch das Auslösen des Events wird der Motor entsprechend angesteuert.
FENSTER_MOTOR_STOP_E	Wird dieses Event ausgelöst, so wird der Motor des Fensterhebers gestoppt.
FENSTER_MOTOR_STOPPED_E	Der Motor des Fensterhebers schickt ein Signal, wenn er stoppt, ohne dass vorher das Signal FENSTER_MOTOR_STOP_E ausgelöst wurde (d.h. der Motor stoppt aus außergewöhnlichen Gründen). Dies ist zum Beispiel der Fall, wenn er wegen eines zu großen Widerstands die Bewegung aus Schutzgründen stoppt.
FENSTER_*x*_C	Diese Bedingung zeigt an, ob das Fenster ganz oben (*x* = OBEN) bzw. unten (*x* = UNTEN) ist. Sie wird entweder durch den Sensor unten in der Fahrertür oder

Kürzel Fensterheber	Beschreibung
	durch die Widerstandsfolie auf wahr gesetzt.
FENSTER_TASTER_*x*_C	Diese Bedingung zeigt an, wenn der Taster des Fensterhebermechanismus in eine bestimmte Richtung gedrückt wird. *x* steht hierbei für HOCH oder RUNTER.

Tabelle 3.4: Beschreibung der Signale für den Fensterheber

3.5 Die Bedienkonsolen beim Fahrer

Beim Fahrer sind links und rechts neben dem eigentlichen Armaturenbrett zwei Bedienkonsolen angebracht, die *Konsole Links* und die *Konsole Rechts*. Diese wurden durch die erweiterte Funktionalität entsprechend verändert.
Abbildung 3.5.1 zeigt die neue *Konsole Links*.

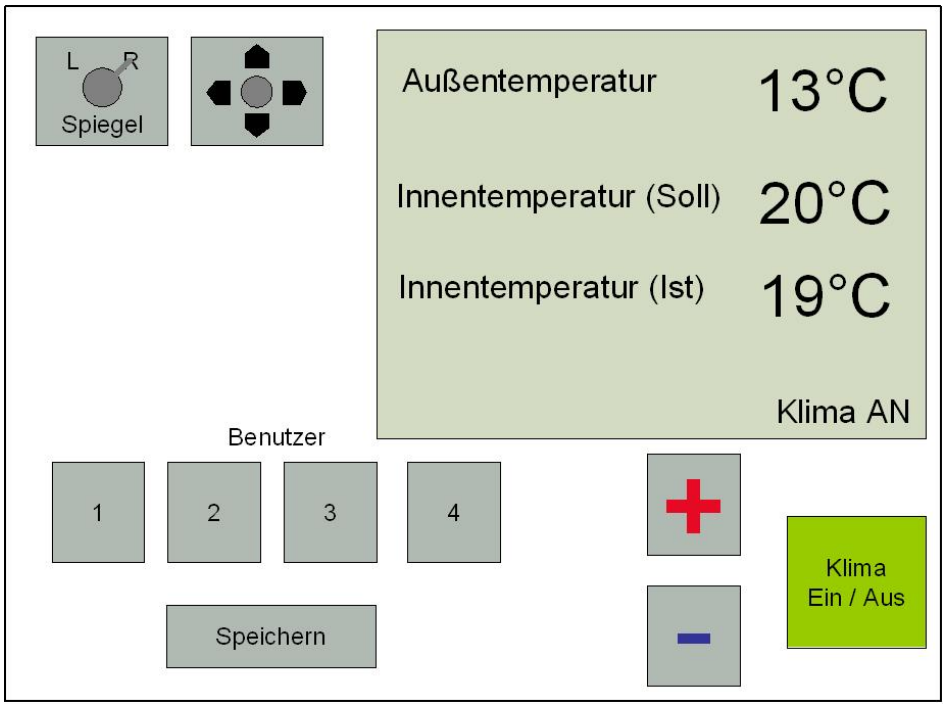


Abbildung 3.5.1: Übersicht über die Bedienkonsole Links

Mit dieser Konsole kann der Fahrer die Außenspiegel des Fahrzeugs einstellen. Hierzu befinden sich oben links die entsprechenden Bedienelemente. Näheres hierzu siehe Abschnitt *3.3 Einstellung der Außenspiegel*.
Rechts unten befindet sich die Einheit für die Steuerung der Klimaanlage bzw. Heizung, welche in Abschnitt *3.1 Steuerung der Klimaanlage* detailliert dargestellt wird.
Das Display oben rechts enthält Informationen für den Fahrer hinsichtlich der Temperatur im Fahrzeuginneren und –äußeren Etwaige Fehlermeldungen die Spiegel oder die Klimaanlage bzw. Heizung betreffend werden hier angezeigt.

Abbildung 3.5.2 enthält die Übersicht über die *Konsole Rechts*.

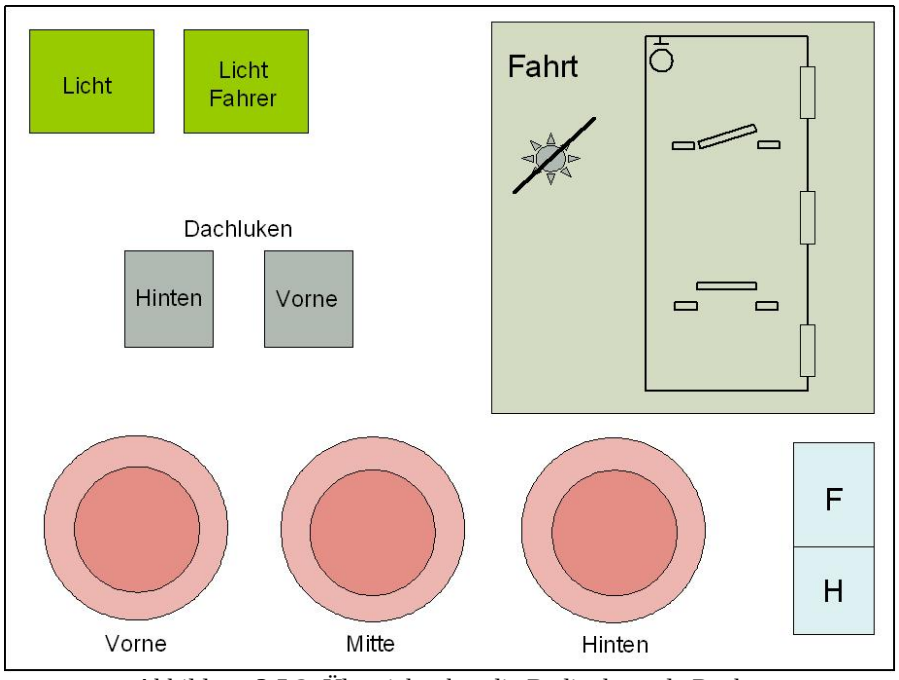


Abbildung 3.5.2: Übersicht über die Bedienkonsole Rechts

Mit der *Konsole Rechts* kann der Fahrer die Türen des Fahrzeugs bedienen.
Darunter sind die Knöpfe für die Steuerung der Dachluken angebracht. Sie werden in Abschnitt *3.2 Steuerung der Dachluken* detailliert betrachtet.

3.6 Das Benutzermanagement

Mit dem Benutzermanagement können bis zu vier verschiedene Fahrer ihre Einstellungen im Fahrzeug speichern und einfach per Knopfdruck abrufen. Das BSG sorgt dann dafür, dass die entsprechenden Positionen der Außenspiegel wiederhergestellt werden.
Das BSG reagiert auf Benutzerwünsche nur, wenn

- die Zündung auf Stellung BETRIEB steht
- die Geschwindigkeit des Fahrzeugs weniger als 10 km/h beträgt

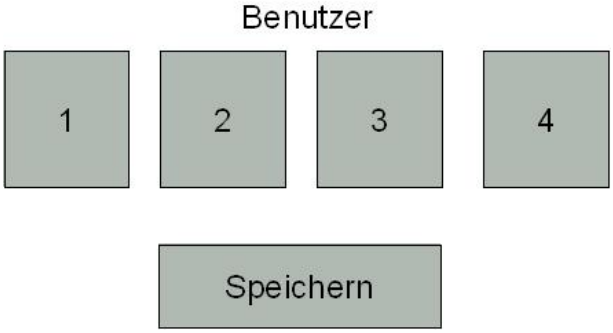


Abbildung 3.6.1: Bedienfelder für das Benutzermanagement

In Abbildung 3.6.1 ist das Bedienfeld für das Benutzermanagementsystem dargestellt. Mit den Zifferntasten 1 bis 4 können vier eingespeicherte Konfigurationen abgerufen werden (im Folgenden auch Speicherplätze genannt). Mit der Taste „Speichern“ kann eine Konfiguration einem bestimmten Speicherplatz zugewiesen werden.
Um einen Speicherplatz zu belegen, muss die Taste „Speichern“ gedrückt gehalten werden und anschließend innerhalb von 2 Sekunden eine der Zifferntasten. Die momentan eingestellten Werte für die Außenspiegel werden dann unter diesem Speicherplatz abgelegt.

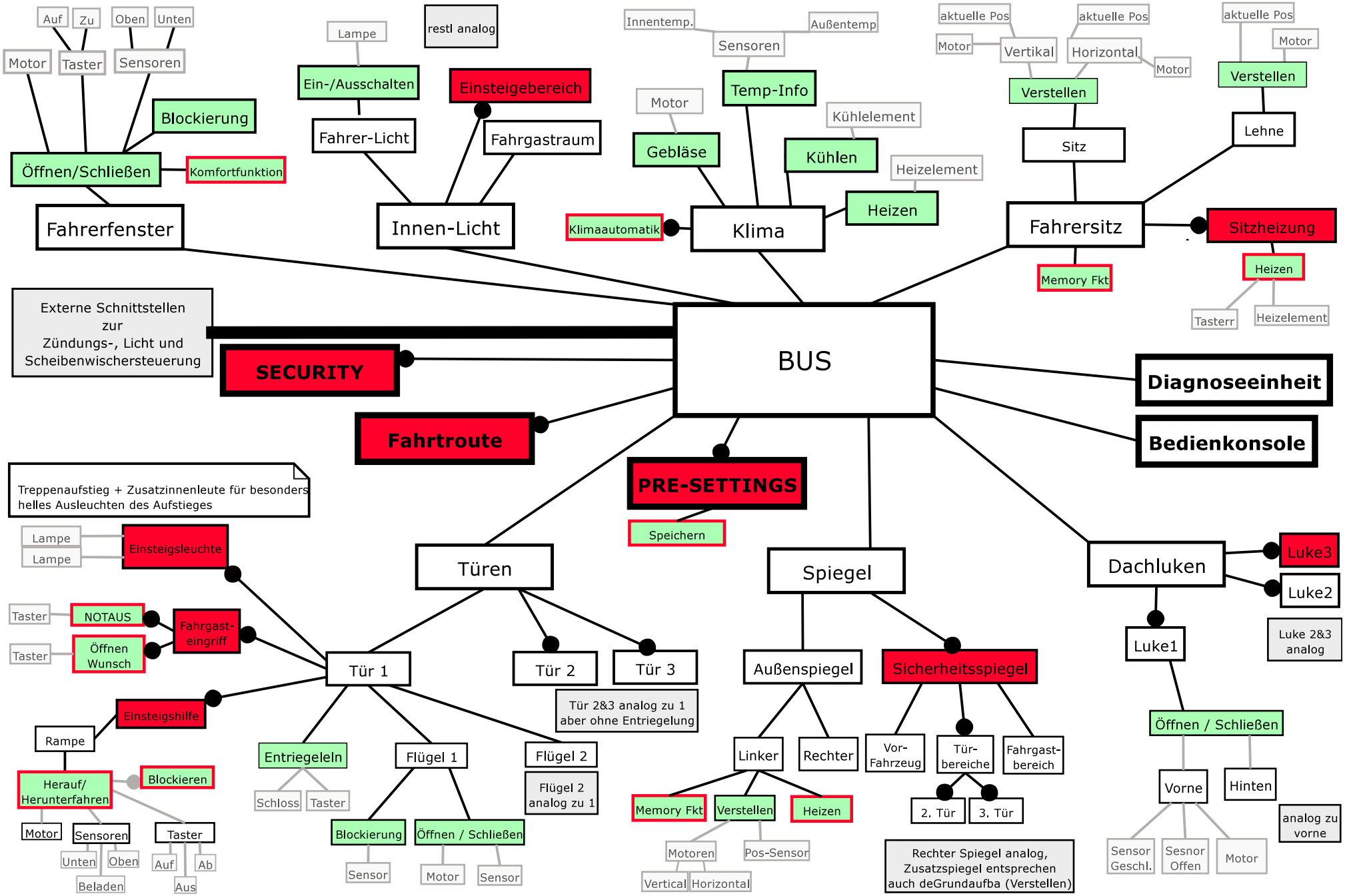
Wird innerhalb dieser 2 Sekunden keine Zifferntaste gedrückt, so muss die Speichern-Taste erst wieder losgelassen werden, um einen erneuten Speichervorgang zu starten. Während die Speichern-Taste gedrückt ist, werden keine Signale der manuellen Außenspiegelverstellung akzeptiert.
Um eine Speicherung abzurufen, muss eine der Zifferntasten für mindestens eine halbe Sekunde gedrückt werden. Wir die Taste danach losgelassen, so fährt das BSG die Spiegel in die abgespeicherten Positionen. Dabei wird folgende Reihenfolge eingehalten:

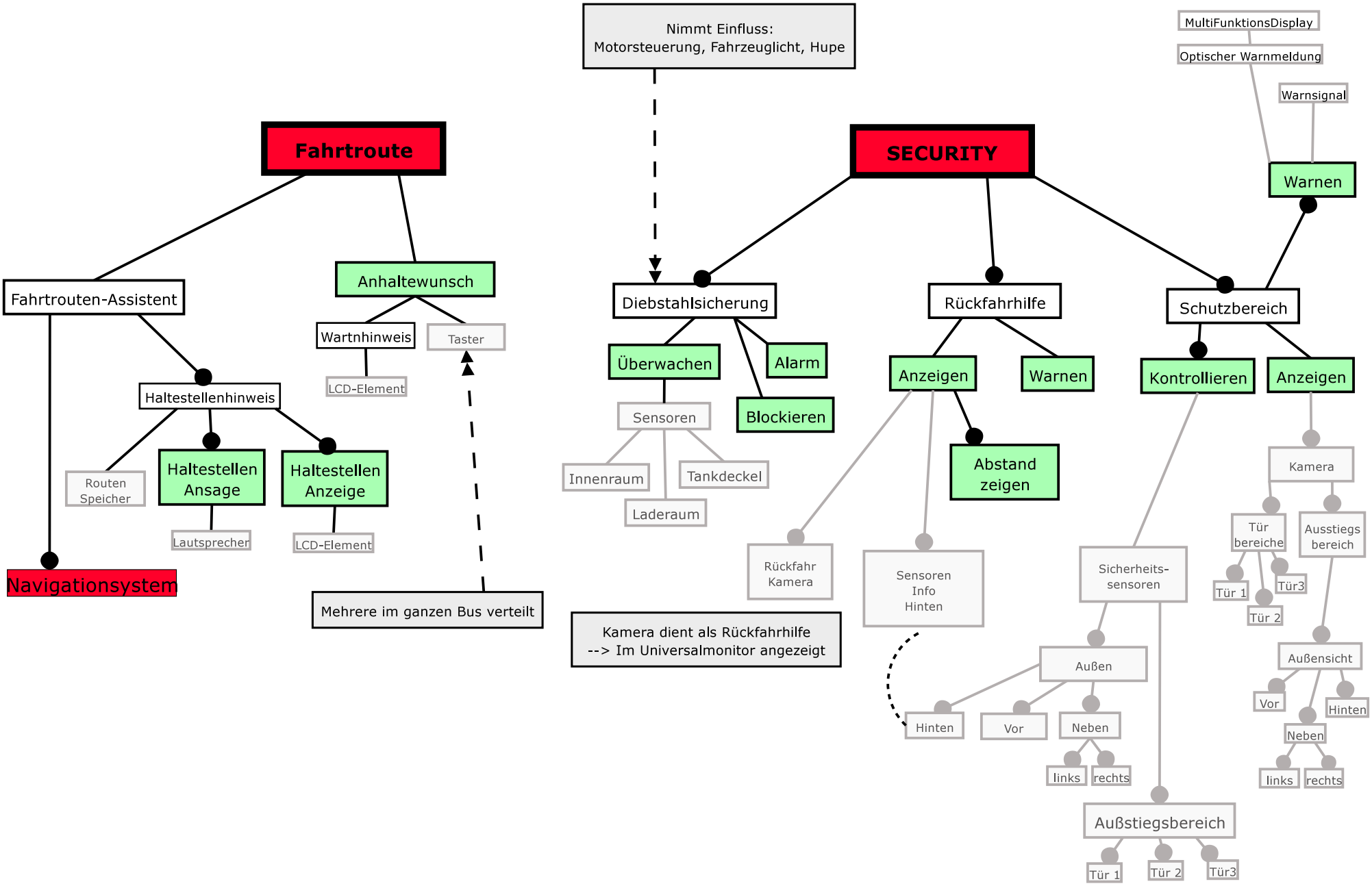
- Zunächst fahren die Spiegel in die Mittelstellung.
- Von der Mittelstellung aus werden die Spiegel anschließend in die Zielpositionen gefahren.

Das Benutzermanagement akzeptiert weitere Befehle durch den Fahrer erst dann wieder, wenn der Einstellungsvorgang abgeschlossen ist. Bis dahin werden alle Tastendrücke ignoriert, auch die Taster für die manuelle Spiegelverstellung werden erst wieder berücksichtigt, wenn die Spiegel in die entsprechende Position gebracht worden sind. Selbstverständlich gelten für die Außenspiegeleinstellungen die gleichen Vorgaben für das BSG wie für andere Benutzer, insbesondere sind alle Sicherheitsaspekte zu berücksichtigen.
Das manuelle Einstellen der Spiegel durch den Fahrer mit Hilfe der entsprechenden Bedienelemente wird sofort abgebrochen, wenn das Benutzermanagement eine gespeicherte Konfiguration einstellt.
Das Einstellen der Positionen durch das BSG im Benutzermanagementbetrieb wird abgebrochen, falls eine der Positionen nicht innerhalb von 20 Sekunden erreicht werden kann. In diesem Fall wird im Display der Konsole Links die Meldung „Benutzermanagement Fehler“ eingeblendet. Der Fahrer kann jederzeit wieder eine Speicherposition abrufen. Gelingt das Einstellen der Positionen, so muss die Meldung im Display verschwinden.
Tabelle 3.6 enthält die für das Benutzermanagement benötigten Signale.

Kürzel Benutzermanagement	Beschreibung
BENUTZ_MGT_POS_*x*_C	Diese Bedingung ist wahr, wenn eine Zifferntaste auf dem Bedienfeld gedrückt wird. *x* steht hierbei für die Ziffern 1 bis 4.
BENUTZ_MGT_SAFE_C	Diese Bedingung ist wahr, wenn die Taste „Speichern“ des Bedienfelds gedrückt wird.
DISPLAY_ERROR_D	String mit einer Länge von 20 Zeichen zur Fehlerausgabe auf dem Display der *Konsole Links*.

Tabelle 3.6: Beschreibung der Signale für das Benutzermanagement


BUS
Externe Schnittstellen zur Zündungs-, Licht und Scheibenwischersteuerung
SECURITY
Fahrtroute
PRE-SETTINGS
Speichern
Diagnoseeinheit
Bedienkonsole
Fahrerfenster
Öffnen/Schließen
Komfortfunktion
Motor
Taster
Auf
Zu
Oben
Unten
Sensoren
Blockierung
Innen-Licht
Ein-/Ausschalten
Lampe
restl analog
Fahrer-Licht
Einsteigebereich
Fahrgastraum
Klima
Klimaautomatik
Gebläse
Motor
Temp-Info
Innentemp.
Sensoren
Außentemp
Kühlen
Kühlelement
Heizen
Heizelement
Fahrersitz
Verstellen
aktuelle Pos
Motor
Vertikal
Horizontal
Sitz
Lehne
Sitzheizung
Heizen
Tasterr
Heizelement
Memory Fkt
Türen
Tür 1
Tür 2
Tür 3
Tür 2&3 analog zu 1 aber ohne Entriegelung
Einsteigsleuchte
Lampe
Treppenaufstieg + Zusatzinnenleute für besonders helles Ausleuchten des Aufstieges
Fahrgast-eingriff
NOTAUS
Öffnen Wunsch
Einsteigshilfe
Rampe
Herauf/ Herunterfahren
Blockieren
Motor
Sensoren
Taster
Unten
Oben
Beladen
Auf
Ab
Aus
Entriegeleln
Schloss
Taster
Flügel 1
Blockierung
Öffnen / Schließen
Sensor
Motor
Sensor
Flügel 2
Flügel 2 analog zu 1
Spiegel
Außenspiegel
Linker
Rechter
Memory Fkt
Verstellen
Heizen
Motoren
Pos-Sensor
Vertical
Horizontal
Sicherheitsspiegel
Vor-Fahrzeug
Tür-bereiche
Fahrgast-bereich
2. Tür
3. Tür
Rechter Spiegel analog, Zusatzspiegel entsprechen auch deGrundaufba (Verstellen)
Dachluken
Luke3
Luke2
Luke 2&3 analog
Luke1
Öffnen / Schließen
Vorne
Hinten
analog zu vorne
Sensor Geschl.
Sesnor Offen
Motor


Fahrtroute
Fahrtrouten-Assistent
Anhaltewunsch
Wartnhinweis
Taster
LCD-Element
Haltestellenhinweis
Routen Speicher
Haltestellen Ansage
Haltestellen Anzeige
Lautsprecher
LCD-Element
Navigationsystem
Mehrere im ganzen Bus verteilt
Nimmt Einfluss:
Motorsteuerung, Fahrzeuglicht, Hupe
SECURITY
Diebstahlsicherung
Überwachen
Alarm
Blockieren
Sensoren
Innenraum
Tankdeckel
Laderaum
Rückfahrhilfe
Anzeigen
Warnen
Abstand zeigen
Rückfahr Kamera
Sensoren Info Hinten
Kamera dient als Rückfahrhilfe
--> Im Universalmonitor angezeigt
Schutzbereich
Kontrollieren
Anzeigen
Sicherheits-sensoren
Außen
Hinten
Vor
Neben
links
rechts
Außtiegsbereich
Tür 1
Tür 2
Tür3
MultiFunktionsDisplay
Optischer Warnmeldung
Warnsignal
Warnen
Kamera
Tür bereiche
Ausstiegs bereich
Tür 1
Tür3
Tür 2
Außensicht
Vor
Hinten
Neben
links
rechts