

Konzeption und Implementierung eines Sachwarmverhaltens von mobilen Kleinrobotern anhand eines Verfolgungsszenarios

STUDIENARBEIT

für die Prüfung zum

Bachelor of Science

des Studiengangs Informatik Studienrichtung Angewandte Informatik

an der

Dualen Hochschule Baden-Württemberg Karlsruhe

18. März 2017

Bearbeitungszeitraum 24 Wochen

Name Manuel Bothner Simon Lang Matrikelnummer 8359139 6794837 Kurs TINF14B2 TINF14B2

Ausbildungsfirma 1&1 Internet SE ifm ecomatic GmbH

Brauerstr. 48 Im Heidach 18

76135 Karlsruhe 88079 Kressbronn am Bodensee

Betreuer Prof. Hans-Jörg Haubner Gutachter Prof. Dr. Heinrich Braun



Unterschrift

Erklärung

Ort, Datum

(gemäß §5(3) der "Studien- und Prüfungsordnung DHBW Technik" vom 29. 9. 2015) Ich versichere hiermit, dass ich die Studienarbeit meiner Studienarbeit mit dem Thema: "Konzeption und Implementierung eines Sachwarmverhaltens von mobilen Kleinrobotern anhand eines Verfolgungsszenarios" selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Ort, Datum	Unterschrift



Abstract



Zusammenfassung



Inhaltsverzeichnis

1	Ein	leitung	S	11
	1.1	Ausga	ngslage	11
	1.2	Zielset	\mathbf{zung}	11
	1.3	Erwar	tetes Ergebnis	11
2	Tec	hnisch	e Grundlagen	12
	2.1	Robot	<mark>ik</mark>	12
		2.1.1	Grundlagen	12
		2.1.2	Mobile Roboter	12
		2.1.3	Antriebsarten	12
		2.1.4	Sensorik	12
		2.1.5	LEGO Mindstorm	12
	2.2	Applie	eation (App) Entwicklung	13
		2.2.1	Native Apps	13
		2.2.2	Web Apps	13
		2.2.3	Hybride Apps	14
		2.2.4	Plattformübergreifende Programmierung	15
		2.2.5	Mono	15
		2.2.6	.Net Framework	15
	2.3	Java		15
		2.3.1	Grundlagen	15
		2.3.2	Java Runtime Environment	15
	2.4	Komm	$egin{array}{llllllllllllllllllllllllllllllllllll$	15
		2.4.1	Grundlagen	15
		2.4.2	TCP/IP	15
		2.4.3	Wifi	15
		2.4.4	Datenaustausch	15
3	The	eoretise	che Grundlagen	15
	3.1	Schwa	rmverhalten	15
		3.1.1	Allgemein	15
		3.1.2	Vorbilder aus dem Tierreich	15
		3.1.3	Szenarien	15
		3.1.4	Algorithmen	15
4	Pro	jektor	ganisation	16
	<i>A</i> 1	Projek	tahlaufulan	16



5	Konzeption				17	
	5.1	Anford	derunsdefinitionen		17	
		5.1.1	Softwarearchitektur		17	
	5.2	Use C	ases		17	
		5.2.1	Connect		17	
		5.2.2	Synchronization		18	
		5.2.3	Szenario		21	
		5.2.4	Exception		21	
	5.3	Komm	nunikation		21	
6	Lös	ösungsansatz				
7	7 Umsetzung					
8	8 Evaluation					
9	2 Zusammenfassung und Ausblick					



Abkürzungsverzeichnis

 $\mathbf{API}\ \mathrm{Application}\ \mathrm{Programming}\ \mathrm{Interface}.$

App Application.



Glossar

Application Eine Application ist ein ausführbares Programm für mobile Geräte, wie Smartphones oder Tablets.

Application Programming Interface Eine API ist eine Programmierschnittstelle, die die Anbindung von Software ermöglicht.



Abbildungsverzeichnis

1	App Entwicklung	13
2	Connect	17
3	Connection	18
4	Synchronization	19
5	Robot list	20
6	Spectator	21
7	Spectator	22
8	Control	23
9	Control	
10	Synchron	25
11	Follow	26



Tabellenverzeichnis



1 Einleitung

Heutzutage werden viele Arbeitsschritte in der Produktion, als auch Dienstleistungen von Maschinen verrichtet, da diese effizienter Arbeiten und weniger Kosten als Menschen verursachen. Da jede Maschine auf einen spezifischen Arbeitsschritt konfiguriert ist, müssen die verschiedenen Maschinen untereinander wie ein Schwarm agieren. Diese Verhaltensstrukturen kommen ursprünglich aus dem Tierreich, wie Fischschwärme, Ameisen oder Bienen. Hierbei erledigt jedes Individuum seine zugewiesenen Aufgaben und hält die anderen Parteien auf dem aktuellen Stand.

In diesem Projekt werden diese Verhaltensmuster aus dem Tierreich aufgegriffen und anhand eines Verhaltensszenarios mit Kleinrobotern verwirklicht, die autonom agieren und kommunizieren, um zusammen ihr Ziel zu erreichen. Dabei sollen Konzepte, sowie Algorithmen für Schwarmroboter entstehen, die auch auf andere Szenarien angewendet werden können.

- 1.1 Ausgangslage
- 1.2 Zielsetzung
- 1.3 Erwartetes Ergebnis



2 Technische Grundlagen

- 2.1 Robotik
- 2.1.1 Grundlagen
- 2.1.2 Mobile Roboter
- 2.1.3 Antriebsarten
- 2.1.4 Sensorik
- 2.1.5 LEGO Mindstorm



2.2 App Entwicklung

Eine App ist ein ausführbares Programm für mobile Geräte, wie Smartphones oder Tablets. Um eine App für ein mobiles Gerät zu entwickeln, müssen wie für andere Anwendungen im Voraus Anforderungen definiert werden, die die Anwendung erfüllen soll. Je nach festgelegten Anforderungen, die an das System gestellt werden, besteht eine begrenzte Anzahl von Möglichkeiten zur Verfügung. Allgemein kennt die App Entwicklung drei verschiedene Arten, die native, web und hybride Entwicklung.



Abbildung 1: App Entwicklung

2.2.1 Native Apps

In der Entwicklung von nativen Apps werden die direkten Ressourcen des Gerätes verwendet. Dazu gehört die Laufzeitumgebung des Betriebssystemes, Bibliotheken und Hardwareschnittstellen. Der Vorteil von einer nativen Entwicklung liegt hauptsächlich darin, dass diese für das Betriebssystem optimiert ist und die vorhandenen Schnittstellen genutzt werden können, um komplexe und rechenintensive Anwendungen zu ermöglichen.¹

Vertreter diese Entwicklung finden sich für verschiedene Betriebssysteme. Der populärste unter ihnen ist bei weitem Android mit einer nativen Java Entwicklung über Android Studio von Google. Sie besitzt aktuellen den höchsten Marktanteil und eine entsprechende Popularität unter Entwickler und Nutzer.

2.2.2 Web **Apps**

Die Entwicklung von web Apps arbeitet mit systemübergreifenden Ressourcen und greift dabei auf gängige Webtechnologien, wie (HTML), (CSS) und JavaScript zurück. Die App wird hierbei nicht wie normale Anwendungen direkt auf dem System des Gerätes ausgeführt, sondern kommt in dessen Browser zur Ausführung. Der Vorteil hierbei ist vor allem, dass diese Art von App auf allen Betriebssystemen lauffähig ist und direkt über das Internet veröffentlicht und aktualisiert werden kann, jedoch wird eine stabile Internetverbindung vorausgesetzt.¹

Von dieser Entwicklung finden sich viele Vertreter mit der Unterstützung diverser Frameworks. Das populärste unter ihnen ist aktuell AngularJS von Google, was auf JavaScript basiert. In Kombination mit anderen Webtechnologien, wie glshtml und CSS lassen sich perfomante web Apps entwickeln.

¹[vgl. 1, Unterschiede und Vergleich native Apps vs. Web Apps]



2.2.3 Hybride Apps

Die Entwicklung von hybride Apps vereinigt die beiden Entwicklungen von native und web. Sie besteht dabei aus einem nativen Rahmen, in der eine web App zur Ausführung kommt, diese besitzt entsprechende Zugriffsrechte auf Hardwareschnittstellen, um diese mit Application Programming Interfaces (APIs) anzusprechen.²

Diese Entwicklung ist aktuell noch sehr jung, jedoch stechen hier bereits verschiedene Vertreter hervor. Der populärste unter ihnen ist Ionic von Drifty, was auf Apache Cordova als Basis zurückgreift. In Kombination mit AngularJS, TypeScript und anderen Webtechnologien lässt sich die web App entwickeln und auf einem beliebigen Gerät unter einem nativen Browser ausführen. Es unterstützt dabei verschiedenste Betriebssystem, wie Android, iOS und Windows. Diese Entwicklungen können dabei meist nicht nur mobil, sondern unter anderem auf weiteren Systemen, wie stationäre bereitgestellt werden.

²[vgl. 2, Native App, Web App und Hybrid App im Überblick]



2.2.4 Plattformübergreifende Programmierung

Xamarin

- 2.2.5 Mono
- 2.2.6 .Net Framework
- 2.3 Java
- 2.3.1 Grundlagen
- 2.3.2 Java Runtime Environment
- 2.4 Kommunikation
- 2.4.1 Grundlagen
- 2.4.2 TCP/IP
- 2.4.3 Wifi
- 2.4.4 Datenaustausch

3 Theoretische Grundlagen

- 3.1 Schwarmverhalten
- 3.1.1 Allgemein
- 3.1.2 Vorbilder aus dem Tierreich
- 3.1.3 Szenarien
- 3.1.4 Algorithmen



- 4 Projektorganisation
- 4.1 Projektablaufplan



5 Konzeption

In diesem Kapitel werden die Anforderungsdefinitionen des Projektes, mit Spezialisierung auf die verschiedenen Use Cases beschrieben.

5.1 Anforderunsdefinitionen

In diesem Abschnitt wird auf die Funktionalitäten und Use Cases des Projektes eingegangen.

5.1.1 Softwarearchitektur

5.2 Use Cases

5.2.1 Connect

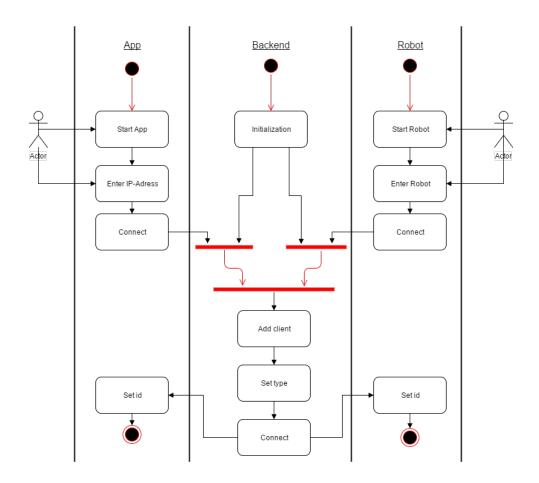


Abbildung 2: Connect

Im Use Case Connect wird eine erste Verbindung durch die Eingabe der IP-Adresse zum Backend aufgebaut. Dabei sendet die Komponente, ob Roboter oder App eine Abbildung seiner selbst als Objekt dem Backend. Daraufhin startet das Backend die Verbindung indem es der Komponente entsprechende Verbindungskommandos zusendet. Sobald eine







Abbildung 3: Connection

Reaktion in einer festgelegten Zeit erfolgt, akzeptiert das Backend die Verbindung und sendet die entsprechende Id für die Komponente. Ab diesem Moment ist die Komponente verbunden und ein Robot für Aktionen entsprechend verfügbar. Die Verbindungsinitialisierung dient hierbei der Verkürzung der Reaktionszeit, die bei einem Roboter sonst entsprechend hoch wäre.

5.2.2 Synchronization

Im Use Case Synchronization werden Daten entsprechend des gesetzten Typen zwischen den Komponenten übertragen. Dabei können einerseits die Roboter als Objekte, oder ganze Szenarien übertragen werden. Dies dient zur Gegenseitigen Synchronisierung der Daten.



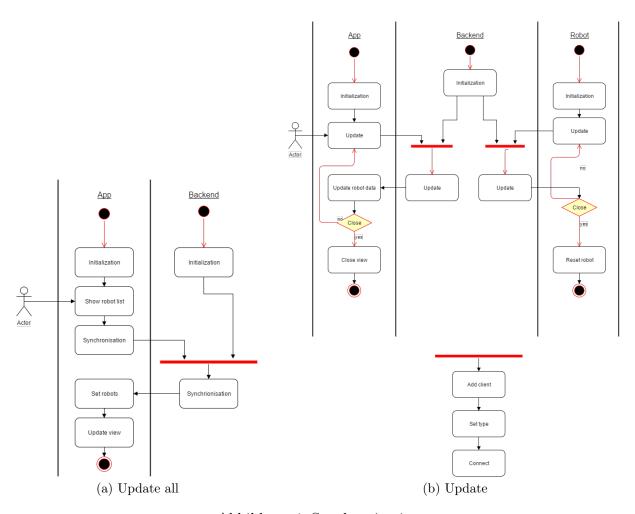


Abbildung 4: Synchronization



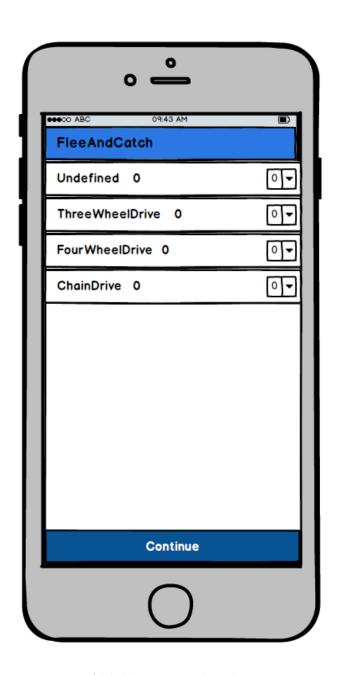


Abbildung 5: Robot list



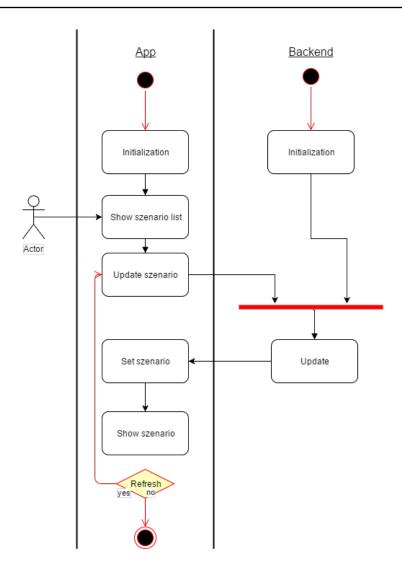


Abbildung 6: Spectator

- 5.2.3 Szenario
- 5.2.4 Exception
- 5.3 Kommunikation



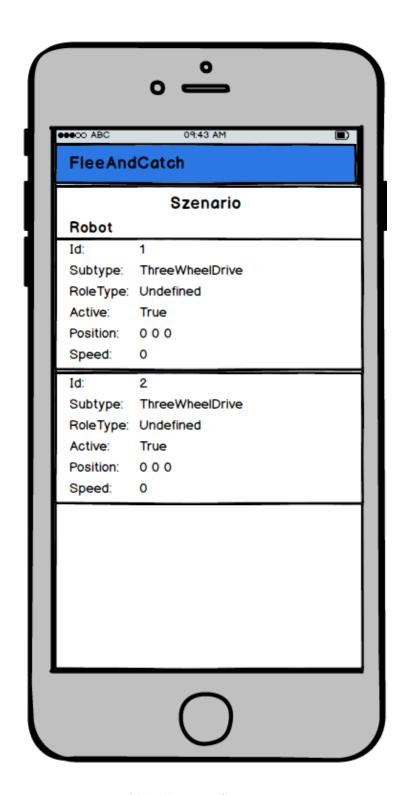


Abbildung 7: Spectator



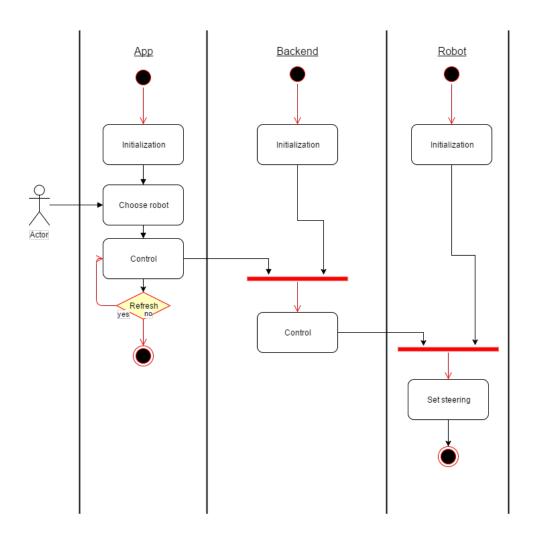


Abbildung 8: Control



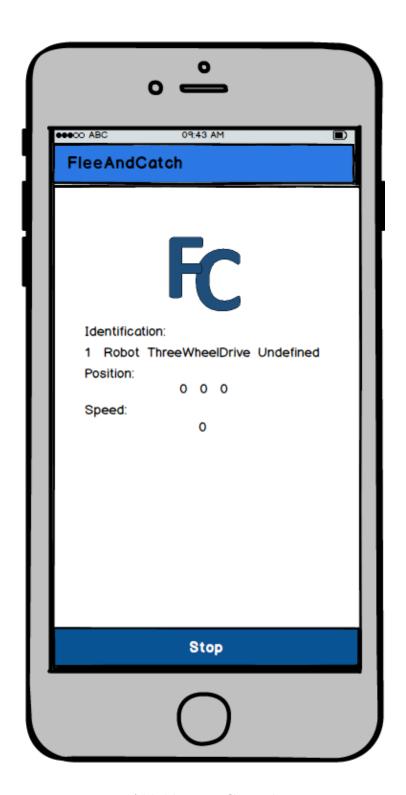


Abbildung 9: Control



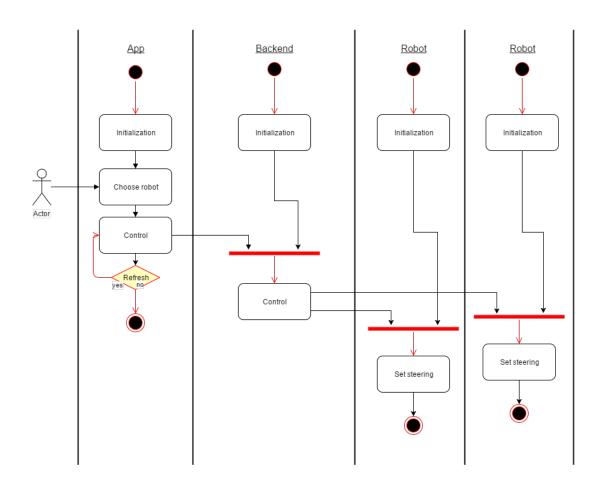


Abbildung 10: Synchron



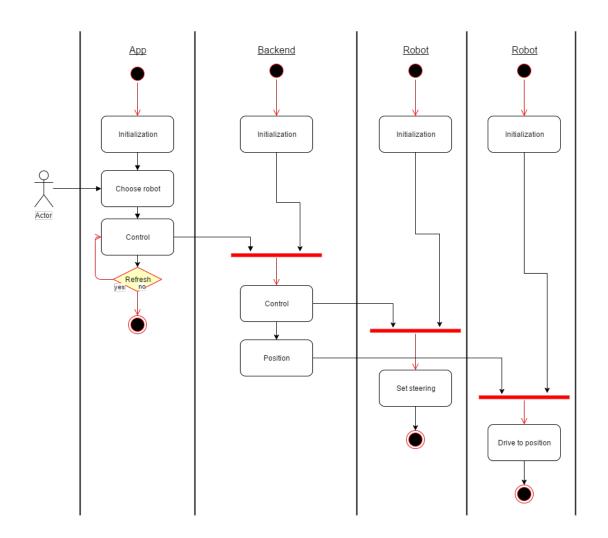


Abbildung 11: Follow



6 Lösungsansatz



7 Umsetzung



8 Evaluation



9 Zusammenfassung und Ausblick



Literatur

- [1] Daniel Würstl. Unterschiede und vergleich native apps vs. web apps. URL http://www.app-entwickler-verzeichnis.de/faq-app-entwicklung/11-definitionen/107-unterschiede-und-vergleich-native-apps-vs-web-apps.
- [2] Petra Riepe. Native app, web app und hybrid app im überblick: Warum native wenn es auch hybrid geht? URL http://www.computerwoche.de/a/warum-native-wenn-es-auch-hybrid-geht,3096411.



Anhang