Gliederung der schriftlichen Arbeit - ALT

1. Einleitung
    1.1. Aufgabestellung und Rahmenbedingung
    1.2. Motivation
2. Grundlagen
    2.1. Stand der Technik
    2.2. Raspberry Pi
        2.2.1. Hardware
        2.2.2. Software
    2.3. Android Smartphone
        2.3.1. Android Betriebssystem
        2.3.2. Konnektivität
    2.4. Android Applikation
        2.4.1. Android Manifest
        2.4.2. Android Activity
        2.4.3. Android Ressource
        2.4.4. Android Layout
        2.4.5. Android View
    2.5. Lösungsansätze
        2.5.1. Android Applikation
        2.5.2. Übertragungsart
        2.5.3. Signalempfang und Auswertung
        2.5.4. Ausführung der Übertragene Signal
3. Vorbereitung
    3.1. Raspberry Pi
        3.1.1. Inbetriebnahme und Vorbereitung
        3.1.2. Werkzeugen
        3.1.3. Installation des Webservers
        3.1.4. Einrichtung des Webservers
    3.2. App Entwicklungsumgebung
        3.2.1. Anlegen des Projekts
        3.2.2. Einbindung Externe Library
        3.2.3. Einstellung der Version Kontrollsystem
4. Implementierung
    4.1. Raspberry Pi Software
        4.1.1. Kommunikation Komponente
        4.1.2. Interpreter Komponente
    4.2. Android App (Pi$Control)

Gliederung der schriftlichen Arbeit - NEU

Inhaltsverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
HTTP	Hypertext Transfer Protocol
XML	Extensible Markup Language
GUI	Graphical User Interface
GPIO	General Purpose Input Output
Raspi	Raspberry Pi
PHP	Hypertext Preprocessor
NFC	Near Fields Communication
ISR	Interrupt Service Routine
WLAN	Wireless Local Area Network
AP	Access Point
CGI	Common Gateway Interface
PWM	Pulse Width Modulation

1. Einleitung

Das Schwerpunktthema des diesjährigen Workshops lautet Eingebettete Systeme. Dies Beschreibt ein in ein umgebendes technisches System eingebettetes und mit diesem in Wechselwirkung stehendes Computersystem. Der Computer übernimmt komplexe Steuerungs-, Regelungs-, Überwachungs- und Datenverarbeitungsaufgaben und verleiht damit dem umgebenden System oft einen entscheidenden Wettbewerbsvorsprung. Die sind heutzutage in alle Bereiche der Technik eingedrungen unter anderen in der Industrie, Automobile, Medizin, und in der Kommunikation. In Moderne Personenkraftwagen der Oberklasse sind beispielsweise zwischen 70 und 80 integrierte und miteinander vernetzte Steuergeräte enthalten (1). Eingebetteten System erfordern in der meistens fällen ein Bedienung- oder Steuerungssystem.

Heutzutage besteht ein Trend, eingebettet System mit Smartphone zu steuern. Dieser Trend kann man besonders in Hausautomatisierung, und Robotik besonders beobachten. In Der Robotik Branche ist die Steuerung von Dronen, Saugroboter und Quadrokopter durch Smartphone immer beliebter. Unterwegs über Internet oder Zuhause Über Wlan werden eingebettet Hausgeräte mit Smartphone gesteuert. Bei einem Smartphone handelt sich um ein Mobiltelefon, dass neben dem Telefonierten noch zahlreiche weitere technische Funktionen aufweist. Dies ist eine Mischung aus Handy, Computer und Spiele-Konsole.

Jedoch ist die Form der Steuerung über Smartphone immer speziell zu einem eingebetteten System, da Sie schon vordefinierte Steuerelemente bieten. Dies hat den Nachteil, dass bei der Entwicklung eines eingebetteten Systems, die Steuerung neu zu Implementierung ist. Um die Entwicklung des Steuerungssystems zu erleichtern, ist es sinnvoll, Ein allgemein einsetzbares Steuerungssystem. Diese Bachelor Arbeit umfasst der Entwurf und die Implementierung ein Solches System für den Raspberry Pi. Sie dient auch als Leitfaden, damit Nachbau und Weiterentwicklung ermöglich wird.

1.1. Problemstellung/Zielsetzung

Ziel dieser Bachelorarbeit, ist die Entwicklung einer Universell Android Applikation für die Bedienung auf Raspberry Pi basierten eingebettet Systeme. Die Applikation überträgt Signal zu den Raspberry Pi über eine Drahtlose Schnittstelle. Der Raspberry soll vorher mit einer Bibliothek und eine API ausgestattet werden.

Aufgabe der Bibliothek ist das Empfange und Weiterleiten von Signale an den API. Letztgenanntes ermöglich dem Nutzer mithilfe vordefinierten Klassen und Methoden auf übertragene Signal zuzugreifen und die Auszuführende Aktion zu bestimmen. Die Bibliothek und Die API sollen deutlich getrennt sein und sich Daten und Nachrichten Austauschen. Die Gleiche Philosophie soll auch bei der Android Applikation verwendet werden.

Die Android-Anwendung besteht aus eine Konfigurierbare Graphische Oberfläche und eine Kommunikation Schnittstelle. Die Oberfläche ist die Benutzerfenster und biete unter anderen Steuerelemente. Der Nutzer soll je nach Wunsch Steuerelemente Einfügen oder Löschen können;

Beim Einfügen bestimmt er die Eigenschaften des Elements. Aufgabe der Kommunikation Schnittstelle ist die Übertragung der von Oberfläche durch Touchscreen Berührung generierten Signal an dem Raspberry Pi. Eine weitere Aufgabe ist das Mitteilen von Raspberry Pi Ausgabe an der Oberfläche. Es sei den auf Android Smartphone oder auf Raspberry Pi sollen eine sollen Komponenten nicht vermischt werden.

Sowohl auf Android als auch auf Raspberry Pi sollen die Komponente deutlich getrennt werden. Zweck ist, dem Nutzer eine große Flexibilität zu bieten und Ihm der Austausch von Komponente zu erleichtern, wenn er sich beispielweise für eine Andere Übertragung Schnittstelle entscheidet.

Die Applikation, seine Bibliothek und API sollen den Idealen Einstiegpunkt für die Entwicklung der Bedienung auf Raspberry Pi basierten eingebettet Systeme über Android Smartphone.

Weiterhin werden alle Schritte beim Entwurf, sowie der Implementierung Dokumentiert, damit zukünftige Nachbau und Erweiterungen unkompliziert gemacht werden können.

1.2. Motivation

Die Motivation sich mit dem Thema dieser Bachelorarbeit auseinanderzusetzen, ist das Anwendung und die Erweiterung der während der Studienzeiten erworbenen Wissen in der Soft- und Hardware. Kenntnissen aus der Software Entwicklung werden angewendet. Es wird Praktisch umgesetzt, wie man ein Projekt vom Grundidee bis zum Implementierung einschließlich Test gestaltet. Die Hauptvoraussetzung für die Implementierung des Android Applikation, ist das Kenntnisse der im Fach Projekt Software Entwicklung kennengelernte Android Framework. Das Raspberry Pi Framework benötigt Kenntnissen vom Raspberry Pi Umwelt, spricht Linux Betriebssystem und seine bekannten GPIO-Pins und deren Programmierung. Grundlagen aus dem Elektrontechnik werden beim Test benötigt. Lichter werden am GPIO-Pins angeschlossen, um die Funktionalität der Applikation zu testen.

Vom Grundidee bis zur Implementierung und Test werden nicht nur im Studium erworbenes Wissen, sondern auch die Ergebnisse eigene Recherche eingesetzt, um diese Bachelorarbeit nach besten wissen und Gewissens abzuschließen.

2. Grundlagen

Dieser Kapitel befasst sich mit der verwendeten Komponente für das Thema dieser Bachelorarbeit. Um unsere Ziel leichter zu erreichen, ist es Wichtig, dass wir das Umfeld beschreiben und die Komponente besser kennenlernen. Der Raspberry Pi, seine Eigenschaften und seine Schnittstelle werden näher betrachtet. Besonders werden die General Purpose Input Output erklärt. Schließlich wird die drahtlose Verbindung Möglichkeiten betrachtet. Danach werden wir uns mit der Android Umgebung beschäftigen. Zuerst werden wir was über eine Android Smartphone erfahren spricht das Betriebssystem, die unterstütze Applikation und seine Konnektivität. Danach wird der Aufbau einer Android Applikation betrachtet. Die unterschiedlichen Komponenten und deren Zusammenhang werden erklärt. Die Nötige Konzepte der Android Framework wie, Manifest, Activity, Lifecycle einer Activity, Layout, Ressourcen, und View werden erklärt.

2.1. Stand der Technik

Vor Anfang dieser Bachelorarbeit, wurde nach Projekten gesucht die das das Thema schon behandeln haben. In diesem Kapitel werden einige davon vorgestellt.

- **PiRelay**[1]

PiReplay ist eine Android App zur Steuerung von Raspberry Pi GPIO-Pins. Die Applikation ist für das Ein- und Ausschalten von GPIO-Pins und kann bis zu fünf Raspberry Pi steuern. Zweck der Entwicklung dieser App, ist die Steuerung Hausgeräten wie Lichter, Ventilatoren, Motoren, Türen und Heizung. Davor muss aber der Raspberry Pi mit einem Webserver und ein PHP Skript ausgestattet sein. Die Oberfläche ist konfigurierbar und kann bis zu 100 Steuerelemente „Relay“ (vgl. Abbildung 1) (2).

Die Berührung eines „Relay“ sendet dem Webserver eine HTTP Anfrage über WLAN. Der Server bearbeitet die Anfrage und leitet die Informationen and dem PHP Skript weiter. Die Informationen enthalten die Pin-Nummer und das Signal. Mittels dieser Informationen, schaltet das Skript der Pin Ein oder Aus (3).

[1] https://play.google.com/store/apps/details?id=com.jasonfindlay.pirelaypro [04.02.2018]

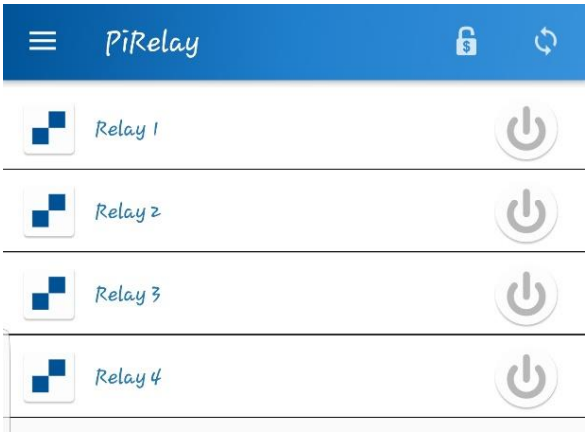

Abbildung 1: Graphische Oberfläche PiReplay

- **BlueTerm[2]**

BlueTerm ist eine Android App zur Kommunikation mit allen Bluetooths serieller Adapter (4). Das Project besteht aus BlueTerm und zwei Python Skripten. Das erste Skript kümmert sich um die Bluetooth Kommunikation und der zweite behandelt der GPIO-Pin. Im Code ist schon festgelegt welchen Pin angesprochen werden soll.

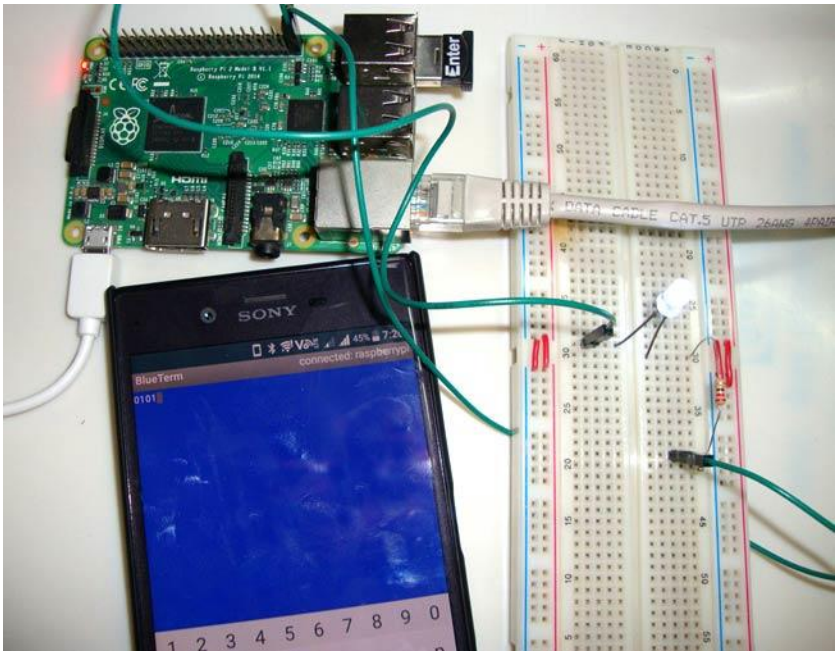

Abbildung 2: Konsolenanwendung BlueTerm

Im Gegenteil zu PiReplay besitzt BlueTerm keine Graphische Oberfläche, sondern eine Konsole für die Eingabe. Durch eingeben von Ein (1) oder Null (0) in der Konsole kann der Pin beziehungsweise Ein- oder Ausgeschaltet werden.

[2] https://play.google.com/store/apps/details?id=es.pymasde.blueterm&hl=fr [04.02.2018]

Aus der Suche ergab sich, dass unsere Thema nicht neue ist. Es existiert bereits viele Projekte die es behandelt haben; sie erfüllen jedoch nicht komplett unser am Anfang festgelegte Voraussetzungen. Zuerst hat der Nutzer kein direkter Zugriff auf die Übertragenen Signale und kann Sie also nicht seinem Wunsch implementieren. Des Weiteren sind Projekt Komponente nicht Austauschbar entweder weil der Quellcode nicht Open Source ist oder keine Sichtbare Grenzen zwischen die Komponenten existiert.

2.2. Raspberry Pi

Raspberry Pi ist ein Einplatinencomputer in Kreditkarten Format. Er wurde von Videospiel-schöpfer David für die Raspberry Pi Fundation entwickelt. In dem Gehäuse steckt ein ARM Prozessor.

Die Intention seiner Entwicklung war ein günstigeres, und einfach zu programmierendes Produkt zu entwickelt. Der Raspberry Pi kostet ungefähr 35 Euro und bietet mit seinem günstigen Preis Kinder, Jugendlichen und Interessierten die nicht über die finanziellen Mittel und sich fürs Programmieren interessieren und begeistert die Möglichkeit einen Computer zu kaufen und sich damit einzuarbeiten (5).

Das Thema dieser Bachelorarbeit wurde mit dem Raspberry Pi 3 Modell B bearbeitet. Er ist seit Februar 2016 verfügbar. Im Lieferumfang befindet sich der Computer und der Stromadapter. Weitere Zubehör muss man sich extra kaufen.

2.2.1. Hardware

In diesem Kapitel werden wir die physische Komponente des verwendeten Modells kennenlernen. Besonders werden wir auf die GPIO-Pins und die Konnektivität der Raspberry Pi 3 Modell B eingehen.

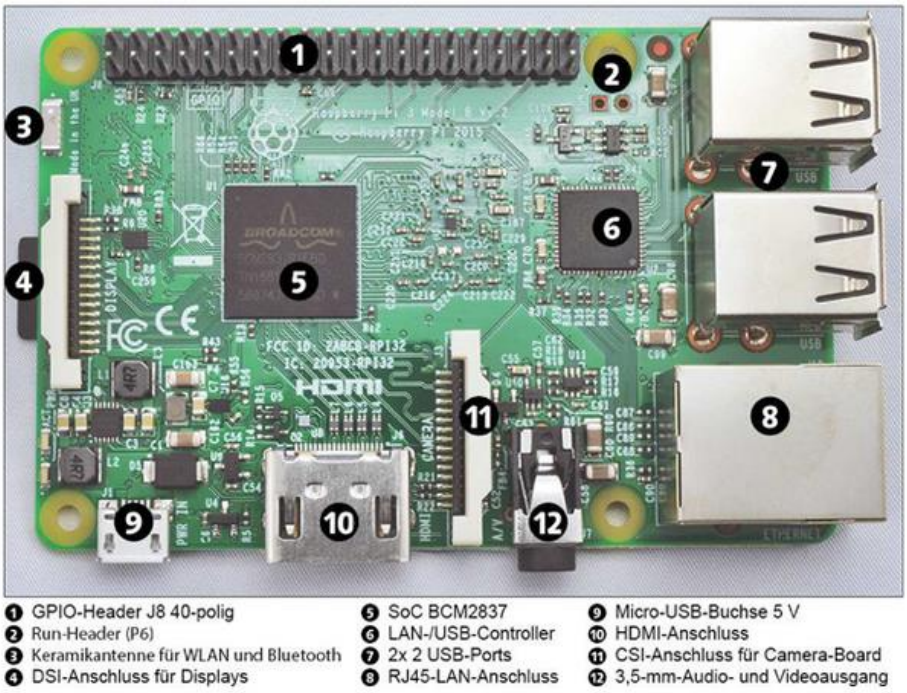

Abbildung 3: Komponentenübersicht des Raspberry-Pi 3 Modell B (5)

Maße (Länge x Breite x Höhe): 85,6 mm x 56,0 mm x 20mm

Gewicht: 40g

SoC: Broadcom-BCM2837

CPU

- Typ: ARM Cortex-A53
- Kerne: 4
- Takt: 1200 MHz
- Architektur: ARMv8-A (64 Bit)

GPU: Broadcom Dual Core Videocore IV

Arbeitsspeicher: 1024 MB

Netzwerk

- Ethernet: 10/100 Mbit/s
- WLAN: Broadcom BCM43143 2,4 GHz WLAN b/g/n
- Bluetooth: 4.1 Low Energy

Schnittstellen

- GPIO-Pins: 40
- CSI, DSI, I^2C, SPI, UART, microSD-Slot
- 4 x USB 2 Port
- DSI Display Port

- CSI Kamera Port

Viedeoausgabe: HDMI (Typ A), Composite Video

Audioausgabe: HDMI (digital), 3,5-mm-Klinkenstecker (analog)

Betriebsspannung: 5 Volt Micro-USB-Anschluss (Micro-USB-B)

- **GPIO-Pins**

Zusätzlich zu den gängigen Schnittstellen einen normalen Computer, besitzt den Raspberry Pi GPIO-Pins. Sie bieten die Idealen Schnittstelle zu eingebettet Welt und können wie Ihren Namen erwähnt, für allgemeine Ein- und Ausgabezwecke verwendet werden. Sie sind auf Abbildung 3 mit der Nummer „1“ bezeichnet. Die folgenden Ausführungen sind wesentlich in den Quellen (7) , [quellen BA Simic] entnommen.

Damit kann man Lichter, Motoren und anderen Geräte und Peripherie an und ausgemacht werden. In Input Modus können Zustände vom einem Taste, Schalter oder Sensor gelesen werden. GPIO ermöglich die Kommunikation über SPI, i²C oder UART Protokoll. Diese Protokolle können sehr leicht in Software implementiert werden. Schließlich können auch PWM (Pulse-Width Modulation) ausgegeben werden. Mit PWM kann man LED Dimmen oder einen Kleinen Digital-Analog Umwandler bauen.

Der Model 3 B besitzt 40 Pins. Sie sind in zwei Reihen von 20 Pins mit einem Abstand von 2,54 mm angeordnet; 28 davon können als GPIO verwendet werden. Die restlichen Pins sind entweder für Erdanschluss oder für Stromversorgung. Zu der Stromversorgung Anschlüsse zählen die 5 V und die 3.3V.

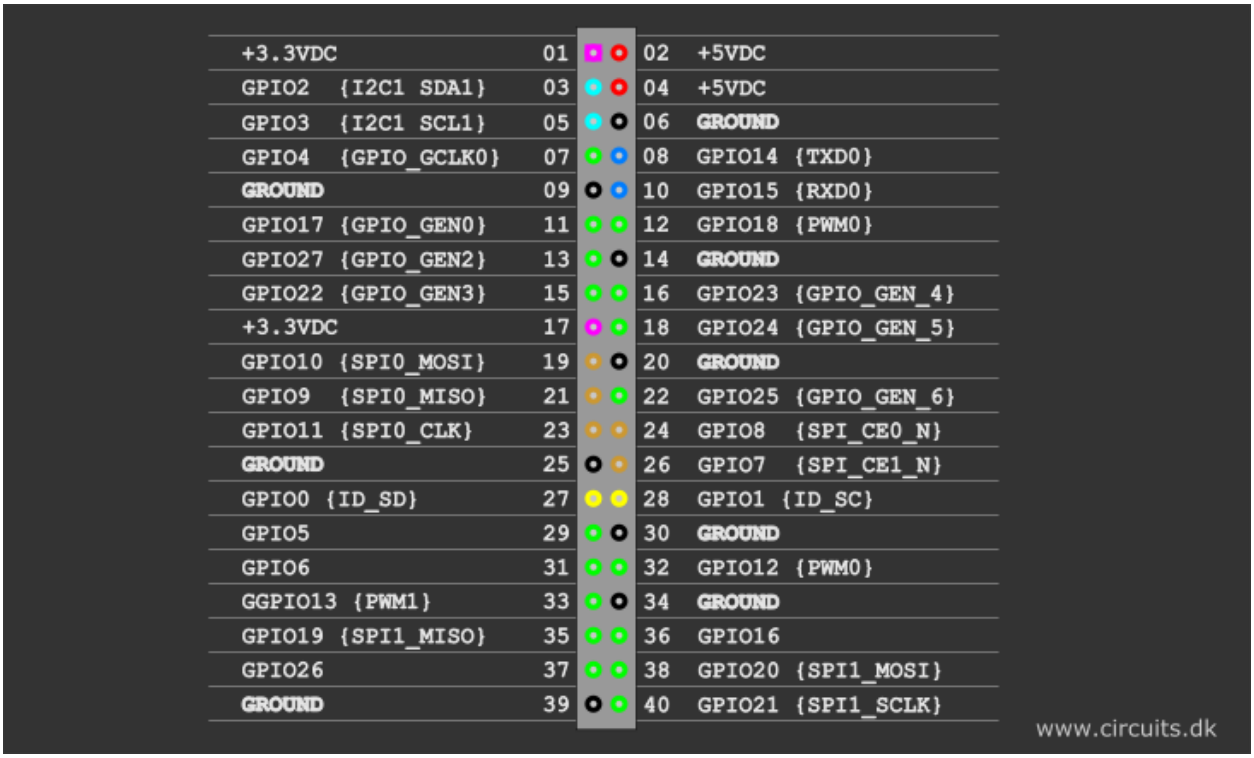

Abbildung 4: Raspberry Pi GPIO-Pins (6)

Es gibt zwei Nummerierungssysteme zur Bezeichnung der Pins. Der BOARD System bezieht sich auf der Physischen Position Pins auf Board. Die Nummer gehen von 1 auf 40 und der Pin mit Nummer 1 steht direkt neben die Bezeichnung „J8“. Der BCM System bezieht sich auf die offizielle Dokumentation des auf dem Raspberry Pi verbauten BCM837-Chips. Dieses System hat der Nachteil, dass sich die Nummer vom Modell unterscheiden. Das System ist auf Abbildung 4 mit der „GPIOxx“ representiert; wobei „xx“ die Nummer bezeichnet. Zur Illustrierung sind die Pins in der folgenden Abbildung farblich in Gruppen markiert, damit Zusammenhängende Funktionen der Pins deutlicher ausgemacht werden können. Die Auswahl eines Nummerierungssysteme ist bei einer Softwaretechnischen Umsetzung erforderlich.

Folgendes Bild verdeutlich die Nummerierungssysteme und gibt mehr technischen Details über die Pins Eingeschalten.

Der maximale Strom ist 50 mA bei der 3.3V Pins und 1 A bei der 5.5V Pins. Die 5V Pins sollten mit sorgfältig behandelt werden, da Sie das Board beschädigen können, wenn Sie mit andere Pins direkt verbinden Sind. Der Pin sollte am besten isoliert werden vor jeder Manipulation.

Ein GPIO-Pin sollte niemals mit einer Spannung größer 3.3V verbunden werden. Dies Konnte das Board beschädigen. Will man ein Pin als Input verwenden, sollte dafür gesorgt werden, dass der maximal Strom nicht überschritten wird; zu diesem Zweck kann einen Externen Widerstand in Reihe angeschlossen werden. Die Empfohlene Grenze ist 0.5 mA. In Output-Modus könne Pins bis zu 16mA Strom fließen lassen.

Die Software Umsetzung der GPIO Pins ist in verschiedene Programmiersprachen möglich. Sehr wahrscheinlich sind Python und C die berühmtesten. Es existiert bereit Bibliotheken in dieser Sprache, die die Implementierung erleichtern; beispielweise RPi.GPIO, Piggpio und WiringPi. Unabhängig von der verwendeten Bibliothek können GPIO-Pins, wenn Sie als Input gesetzt sind, entweder in Interrupt- oder in Polling-Modus gesteuert werden. Beim Polling fragt man Signale regelmäßig innerhalt einer Schleife ab und wertet diese Signale aus. Ein Großer Nachteil dieser Modus ist, dass auf Signale nicht sofort reagiert werden kann, sondern erst mit einer Verzögerung. Bei Interrupt wird sofort reagiert. Wenn ein Interrupt Signal auftritt, wird der aktuelle Befehl des Hauptprogramms beendet, die nötigen Register auf dem Stack gerettet und dann anschließend sofort die „Interrupt Service Routine“ aufgerufen. Dieser Modus hat also kaum Zeitverzug zwischen der Ausführung der ISR und dem Auftreten des Signals; Sie ist für zeitkritischen Signal geeignet. SPI, I²C oder UART Kommunikationsprotokolle können auch einfach in Software implementiert werden.

Es ist wichtig zu wissen, dass die Raspberry Pi GPIO für Echtzeit oder Zeitnahreagierende Systeme nicht richtig geeignet sind, da der Betriebssystem konnte jeder Zeit ein anderer Prozess eine höhere Priorität vergeben.

- **Konnektivität**

Die Konnektivität eines Gerätes Umfasst die Schnittstellen, die das Gerät mit der Welt verbunden kann. Das Thema dieser Bachelorarbeit beinhaltet das Wort „drahtlose Übertragung“; Es ist besonders Wichtig, dass wir die drahtlose Schnittstelle der Raspberry Pi näher betrachten.

Der Model 3 B beinhaltet einen eingebauten Bluetooth und WLAN und müssen nicht mehr zwingend über zusätzliche USB-Adapter installiert werden.

Der Bluetooth Chip arbeitet mit BTLE, auch bezeichnet als Bluetooth 4.1 Low Energy. Damit bringt der Raspberry Pi 3 Model B

Das WLAN Modul unterstützt die Standards 802.11b, g und n und arbeitet im 2.4 GHz Band (BCM43143). Das Modul hat eine maximale übertragungsrate von 150 Mbit/s.

2.2.2. Software

- **Betriebssystem**

Das Betriebssystem ist eine Zusammenstellung von Computerprogrammen, die Systemressourcen eines Computers wie Arbeitsspeicher, Festplatten, Ein- und Ausgabegeräte verwaltet (8). Es gibt zwei Hauptkategorie vom Betriebssystem für den Raspberry Pi: Die Eingebettet und die Mehrzwecksysteme.

Die Eingebettet sind für bestimmte Zwecke entwickelt. Sie besitzen meistens keine graphische Oberfläche. Durch Ihrer geringe Anforderungen in Bezug auf Energiebedarf sowie Arbeits- und Massenspeicher eignen sich eingebettet Betriebssysteme für Zeitnah reagierenden System wie Industrie Anlagen, Dronen, Antiblockiersystem und Roboter. Ein Beispiel wäre Minoca OS.

Die Mehrzweckssysteme dagegen brauchen viele Speicher und Energie und haben längere Reaktionszeit. Wie Ihren Namen es erwähnt, eignet sich diese Systeme für Computer gängige Aufgaben. Sie haben längere Reaktionszeit, brauchen viele Speicher; deren Vorteil ist die einfache Handhabung. Sie besitzen meisten eine Schöne graphische Oberfläche. Auf Internet sind mehrere Mehrzweckssysteme zu finden. Die Üblichen sind Ubuntu, Noobs und Raspbian.

Das beliebteste Betriebssystem ist der kostenlose auf Debian Linux basierte und für Raspberry Pi Hardware optimierte Raspbian. Seine Beliebtheit bedankt seine kostenlose Lizenz, seine schnelle Leistung, Seine vorinstallierte Software und Tools.

Raspbian kommt mit mehr als 35.000 Software Pakete und ist offiziell unterstützt von der Raspberry Fondation.

Besorgt sich man ein Bildschirm, eine Maus, eine Tatstatur und Ihre nötigen Kabel, kann man die an dem Board anschließen und den als vollständige Ersatz für einen Computer Nutzen. Der Raspberry Pi reicht mit seinen Eigenschaften vollkommen aus für gängige Computer Aufgaben. Damit kann man zum Beispiel Problemlos im Internet surfen, Text Schreiben Music Spiele, oder Einige Kleine Spiele Spielen.

Zu diesem Zweck wird das Linux Betriebssystem mit einige vorinstalliert Software geliefert. Es kann unter anderen Chromium für Internet, Leafpad und LibreOffice für Textverarbeitung. Für Bild—und Videoverarbeitung kann der Raspberry und ressourcenintensive spiele kann der Computer überfordert werden. Der Nutzer kann Selbstverständlich auch seine Eigene Programm installieren. Dafür bietet Der Raspberry Pi Repository eine unbegrenzte und abwechslungsreiche Auswahl. Gemäß der Grundidee der Entwicklung der Raspberry Pi, werden auch Entwicklungstool in Betriebssystem vorinstalliert. Man kann unter anderen GenyMotion, Python, Greenfoot Java, Scratch und Bluej Java.

Es ist auch möglich Android und Windows auf Raspberry Pi zu installieren.

- **Anwendung**

In diesem Kapitel werden wir erfahren, welche Programmiersprachen von Raspberry Pi unterstützt sind. Das Betriebssystem hat die Aufgabe, die Verbindung zwischen den materiellen Ressourcen, dem Benutzer und den Anwendungen sicherzustellen. Der Nutzer kann je nach seinem Wünschen entweder zusätzliche Programme installieren oder seine eigenen Anwendungen Schreiben. Da

Der Name des Raspberry Pi knüpft an die Tradition an, Computer nach Früchten zu benennen. Bekannte Vertreter sind Apple und Blackberry. Der Zusatz „Pi“, angesprochen wie das englische Wort „Pie“, übersetzt für Tortenstück, wird oft fälschlicherweise als Solches interpretiert, steht jedoch für Python Interpreter, eine Andeutung für die Hauptprogrammiersprache des Raspberry Pis.

Bei Python handelt es sich um eine höhere Programmiersprache, die sich durch ihre Komplexität der Maschinensprache entfernt. Erst der Einsatz von Interpreter oder Compiler Übersetzt Befehle

des Programmiercodes in Maschinensprache, die der Mikroprozessor bzw. Mikrocontroller versteht.

Der Raspberry Pi kann aber nicht nur mit Python Programmiert werden. Er unterstütz auch Scratch, C, C++, Java, Perl und Skriptsprachen wie HTML5, PHP, JavaScript. Da der Raspberry Pi auf einem Linux-Kernel operiert, können über die angebotenen Paketquellen nach Belieben weitere Programmiersprachen hinzugefügt.

Mit Hilfe einer sinnvollen Programmiersprache wird der Raspberry Pi Framework entwickelt. Der Auswahl wird in den Nächten Kapitel Diskutiert.

2.3. Android Smartphone

Wie eingangs erwähnt, ist ein Smartphone ein Mobiltelefon, das umfangreiche Computer-Funktionalitäten und Konnektivitäten zur Verfügung stellt. Bei Android Smartphones handelt sich um Smartphone, die mit dem Android Betriebssystem ausgestattet sind.

2.3.1. Hardware

Wegen Vielfalt von Android Smartphone, ist es schwer seine Physische Komponente ins Details zu beschreiben. Laut Statista[3] gibt heutzutage 2,32 Mrd. Smartphone Nutzern weltweit. Sie sind je nach Hersteller und Modell unterschiedlich gebaut. Jedoch teilen sich Smartphone einige Eigenschaften. Wir können unter anderen Prozessor, Speicher, Touchscreen, Batterie und Kamera nennen. Zu der gemeinsamen Funkschnittstelle gehören WLAN, und Bluetooth.

Moderne Smartphone besitzen neue Technologie wie NFC, A-GPS, Glonass, und Galileo. Sie sind auch mit Vielen Sensoren ausgerüstet wie Gyroskope, Barometer, Magnetometer (8).

2.3.2. Software

- **Betriebssystem**

Android ist ein Betriebssystem für Smartphone, Tablets und mehr. Sie wurde von Open Hanset Alliance unter Führung von Google entwickelt. Er zählt zu den drei wichtigsten Mobilen-Betriebssystemen . Die erste Android-Version kam im September 2008 auf dem Markt und hat sich sehr schnell verbreitet (10). Laut Google wurde im September 2012 täglich 1,3 Millionen Android-Geräte aktiviert und fast 60 Prozent aller mobilen Endgeräte dürfen mit Android ausgeliefert werden (8).

[3] https://de.statista.com/themen/581/smartphones/ [06.02.2018]

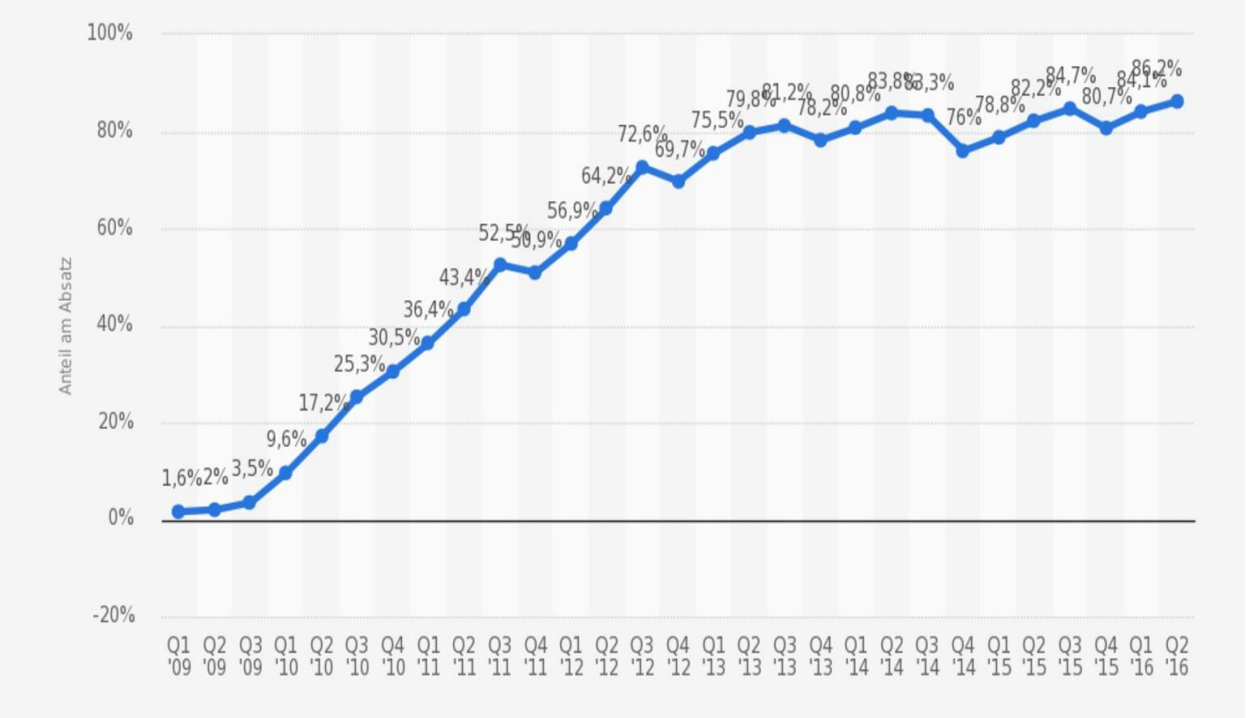

Abbildung 5: Markanteil vom Android Smartphone weltweit vom 2009 bis 2016 (7)

Das Betriebssystem beinhaltet vier Schichten und Jeder abstrahiert die darunterliegende. Er besteht aus ein angepasster Linux-Kernel. Das Kernel bildet die Schicht zwischen der Hardware und der restlichen Architektur und enthält unter anderem Hardware und der restlichen Architektur und enthält unter anderem Hardwaretreiber, Prozess- Speicherverwaltung sowie Sicherheitsverwaltung. Android beinhaltet eine Reihe von Bibliotheken für einige Kernkomponenten, die aus Performancegründen in C und C++ programmiert sind. Sie stellen die meisten Funktionen zur Verfügung, wie z.B. Medienvergabe, den Surface Manager, und die Standard C Bibliotheken.

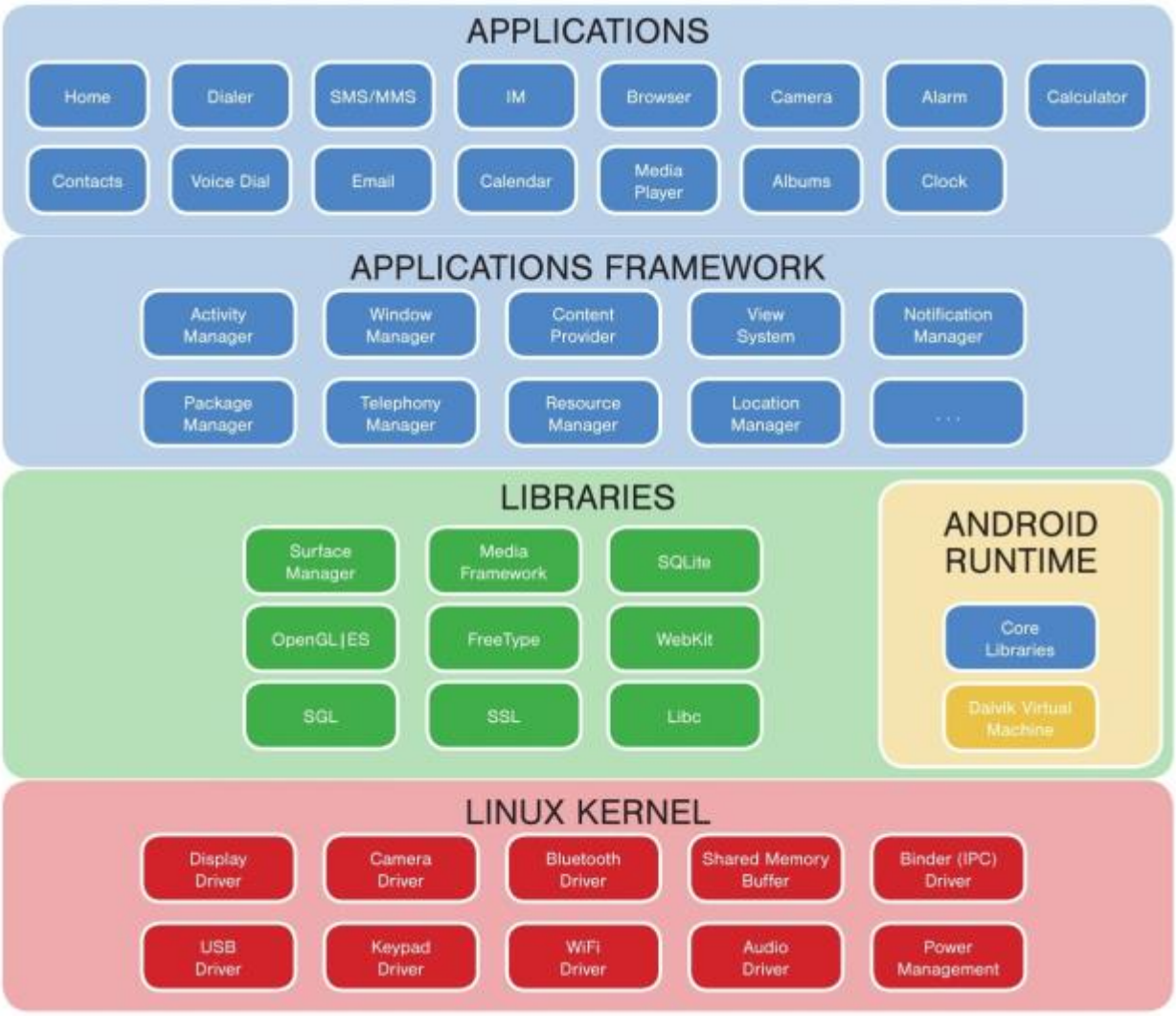

Abbildung 6: Architektur Android (11)

Da Android ein Multiprozess Betriebssystem ist, läuft jede Anwendung in Ihrer eigenen Instanz der Virtuelle Maschine. Dieser Umstand ist vor allem für die Sicherheit und die Anwendungsentwicklung von Bedeutung. Die Virtuelle Maschine wurde speziell dafür entwickelt mehrere virtuelle Maschinen effizient parallel ausführen zu können. Die Ausgeführten Datei sind in für einen minimalen Speicherverbrauch optimierten dex-Format. Dieses Format wird, nachdem sie von Java Kompiliert wurden, mit dem dx-Werkzeug in das nötige Format umgewandelt. Die Virtuelle Maschine benötigt weniger zwischenschritte um den Bytecode auf dem Prozessor auszuführen, weil das Register nicht stapelbasiert ist. Die Maschine nutz die Low-Level-Speicherverwaltung und das Threading des optimierten Kernels aus (12).

Das Anwendungsframework ist die Schicht, die für die Entwicklung von Android Applikation von Bedeutung und stellt den Rahmen für eine einheitliche Anwendungsarchitektur bereit. Ziel ist es, Anwendungen nach einheitlichen Richtlinien zu entwickeln und somit die Integration und Wiederverwendung von Anwendungen unter Android zu vereinfachen (12). Es erlaubt den

Entwicklern die Benutzung von bereits vorhandenen Elementen zu Erstellen von „Graphical User Interface“ (GUI) oder die Nutzung von Ressourcen. Hintergrunddienste, Nachrichtenaustausch mit anderen Applikationen und das Nutzen von Hardware Ressourcen wird ermöglich. Der Zugriff auf dieser Ressource muss ebenfalls vom Nutzer bei der Installation der Applikation oder während der Laufzeit erlaubt zugestimmt werden, damit vermeidet man unerlaubte Zugriffe.

Android-Betriebssystem beinhaltet grundlegende Apps wie E-Mail, SMS-, Kalender-, Karten-, Kontakte-, und Browseranwendungen. Diese Anwendungen wurden mit der Programmiersprache Java implementiert.

2.3.3. Android-Anwendung

In diesem Kapitel werden wichtigsten Komponente eine Android-Anwendung und deren Zusammenhang kennengelernt. Die App besteht grundsätzlich aus einem Manifest, den Java Code und Ressourcen. Der Java Code beinhaltet Activities. Ressourcen bestehen aus Layouts (Zusammenfassung von Views), und notwendige Bestandteil des Programms wie Zeichenfolge, und Binärdateien. Der Java Code und die Ressourcen sind im Projekt unter den Ordner *java* beziehungsweise res zu finden. Der Java Code beinhaltet auch die Programmlogik und die Ressourcen sein Erscheinungsbild. Das Verständnis von Vererbung, Interface, Threads, Abstrakte Klasse in Java sind Notwendig für die Entwicklung Android Anwendungen.

Die Hauptquelle diesen Ausführungen ist (15).

Abbildung 7: Aufbau Android-Anwendung

Android Manifest

Wie auf Abbildung 7 zu sehen ist, gehört der Manifest zur Darstellung-Kategorie. Sie beinhaltet grundlegende Informationen, die das System brauch, um die Applikation zu Starten und beschreibt seine Komponenten. Er beinhaltet die Liste von Aktivitäten und mit welche die Anwendung starten soll, Zugrifferlaubnisse, die die Anwendung umfassen. Der Manifest ist in der Auszeichnungssprache XML geschrieben.

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="testapp.android.test">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.CAMERA" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Test"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Abbildung 8: Beispiel Android Manifest

Zugriff auf System Ressourcen wird mit „uses-permission“ bekannt gemacht werden; Mit dem Attribut „android:name“ wird angegeben, auf welche Ressource man zugreifen will. Das Auflisten von Aktivitäten die der Anwendung umfassen werden mithilfe „activity“ gemacht. Übrigens müssen Sie immer Kinderelemente von „application“ sein.

Activity

Eine Android-App besteht im Wesentlich aus Activities (3).Sie ist ein Fenster mit dem der Nutzer mittels Tasten- oder Touchscreen Berührungen interagiert. Das Aussehen der Benutzerfenster wird mit Layouts definiert. Es kann immer nur eine Activity gleichzeitig aktiv sein, bei einem Wechsel wird die zuvor laufende pausiert. Das Aufrufen erfolgt über Intents[4]. Intents sind Objekte, die Informationen über eine Operation beinhaltet und das Bindeglied zwischen mehreren Komponenten von Android bilden.

[4] https://developer.android.com/reference/android/content/Intent.html [06.02.2018]

Im Gegenteil zu anderen Plattformen, bei denen man Anwendungen nur starten oder beenden kann, hat Android mehrere Status, die eine Anwendung in Ihrer Laufzeit durchläuft (vgl. Abbildung 9)

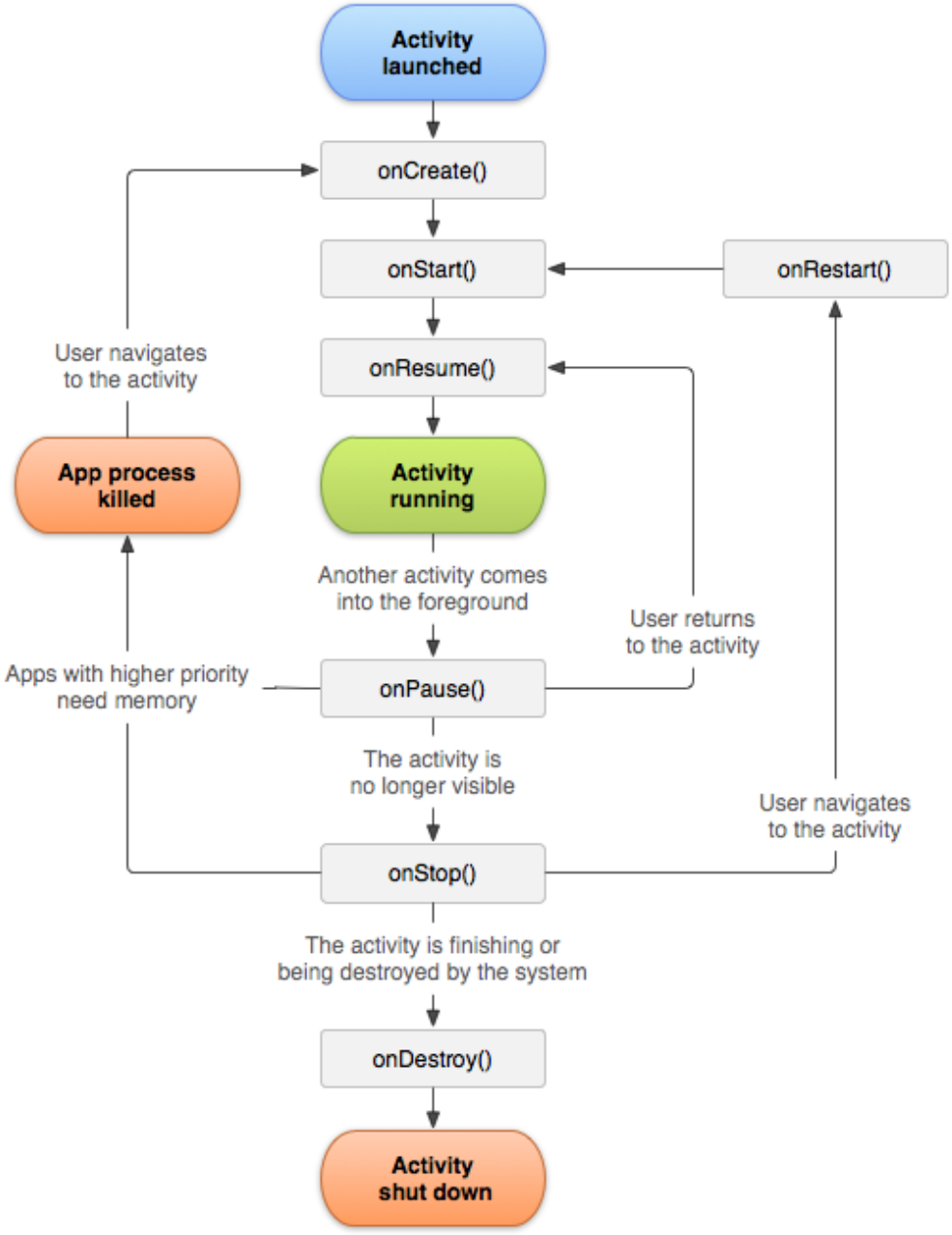

Abbildung 9: Zustandsdiagramm einer Android Activity (16)

Android bietet den Nutzer viele Callback-Methoden zur Verwaltung der Lebenszyklen von Activities. Das aufrufen dieser Methoden wird bis auf die *onCreate*-Methode vom System Übernommen. Beim Starten einer Anwendung und somit der ersten Activity, werden gleich drei

Methoden Aufgerufen: *onCreate, onStart,* und *onResume.* Die wesentliche Logik der Activity soll in der *onCreate*-Methode implementiert werden. Da wird auch mithilfe von *setContentView*-Methode bekannt gegeben, welches Layout (GUI) gehört zu der Activity. Dieser Methode muss ein Layout übergeben werden.

```java
package testapp.android.test;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Hier kann der Nutzer die Aktivitätslogik implementieren.
    }
}
```

Abbildung 10: Minimal Beispiel einer Android Activity

Layout

Ein Layout bestimmt die Struktur eines GUIs. Sie bestimmt die Position der Oberfläche Element, deren Abstände, und deren Attributen. Es gibt verschiedene Typen von Layout. Wir können unter anderen *LinearLayout*, *RelativeLayout*, *TableLayout*, *GridLayout.* Der wesentliche Unterschied ist, die Anordnung der Oberfläche Elementen. Ein LinearLayout positioniert beispielweise die Elemente in eine Reihe.

Layouts können sowohl in XML als auch in Java deklariert werden. Android bietet ein einfaches XML Vokabular zur Erstellung von Layouts. Das Layout wird dann unter Ressourcen Folder abgelegt. Weiterhin gibt auch die Möglichkeit durch Vererbung einer *ViewGroup* Unterklasse, ein Layout in Java zu erstellen. Zu jedem Layout Type gehört eine Java Klasse. Die Positionierung des Elements und das Layout Große wird dann mithilfe der *LayoutParams*-Klasse bestimmt (17). Layouts können auch geschachtelt sein.

```java
import android.content.Context;
import android.util.AttributeSet;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.RelativeLayout;

public class MyRelativeLayout extends RelativeLayout {

    public MyRelativeLayout(Context context) {
        super(context);
        init();
    }

    public MyRelativeLayout(Context context, AttributeSet attrs) {
        super(context, attrs);
        init();
    }

    private void init() {
        setLayoutParams(layoutParams());
    }

    private LayoutParams layoutParams () {
        LayoutParams params = new LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
                              ViewGroup.LayoutParams.WRAP_CONTENT);
        return params;
    }
}
```

Abbildung 11: Minimal Beispiel eines Android-Layout in Java

Ein Layout besteht im wesentlich aus Bedienelemente. In Android bezeichnet man Elementen das Layout umfassen als *View.*

View

Views sind Steuerelemente von Android GUI. Sie reagieren auf Touch- oder Tatstatur-Ereignissen und ermöglichen somit den Nutzer mit der Anwendung zu Kommunizieren. Android bietet verschiedene Typ von Views, die sich von Ihren Aussehen und Bedienungsform unterscheiden. Jeder View-Typ bietet Methoden und Interface zur Verwaltung von Ereignissen.

View-Klasse	Beschreibung
TextView	Einfache Textausgabe
Button, ImageButton	Schaltfläche
EditText	Texteingabe
CheckBox	Ankreuzfeld
RadioGroup, RadioButton	Auswahlschalter; einer aus RadioGroup
Spinner	Auswahlliste
DatePicker, TimePicker	Auswahl von Datum und Zeit
AnalogClock, DigitalClock	Anzeige Uhrzeit (analog, digital)

Abbildung 12: Beschreibung einige Android-Views (18)

Views können wie Layouts sowohl in XML oder in Java erstellt werden. Die Erstellung einer View in Java erfordert die Vererbung einer Unterklasse vom *View*.

```java
package testapp.android.test;

import android.content.Context;
import android.util.AttributeSet;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;

public class MyView extends EditText {

    public MyView(Context context) {
        super(context);
    }

    public MyView(Context context, AttributeSet attrs) {
        super(context, attrs);
    }
}
```

Abbildung 13: Erstellung einer Android-View in Java

3. Lösungsansatz

Diese Kapitel umfasst die Entscheidungen bezüglich Aufbau und Implementierung der Android-Anwendung und Sein Raspberry Pi Teil, bezogen auf die Grundlagen, und dass damit einhergehende erlangte Wissen, die dann in Entwurf und Implementierung einfließen. Der Komplette Aufbau wird mithilfe von Bilder und Graphen Skizziert. Weiterhin wird auch auf jeder Komponente Zugegriffen und die Entscheidung über seiner Implementierung begründet.

3.1. Konzept

Die wesentlichen Herausforderung dieser Bachelor Arbeit ist die Entwicklung einer Modular und Universelle (Wiederverwendbar) Android Anwendung zur Bedienung auf Raspberry Pi basierten eingebettet Systeme. Beim Entwurf wurde auf diese Kriterien besonders beachte und ein Konzept entwickelt der sie erfüllen soll. In den Nächsten abschnitten wird die einzelnen Kriterien eingehen, die das Konzept umfassen.

- **Modularität**

Die Modularität (auch Baustein- oder Baukastenprinzip) ist die Aufteilung eines Ganzen in Teile, die als Module, Komponenten, Bauelemente oder Bausteine bezeichnet werden und zusammen interagieren können (19). Um diese Zwecke zu erfüllen, wurden Vollständig implementierten Module definiert. Bei Module handelt es sich in unseren Kontext, um Elemente die sich für gemeinsame Zwecken entwickelt wurden. Dadurch entstand auf die Android Anwendung zwei Modulen: die GUI und die Kommunikation Schnittstelle. Die Anwendung-GUI ist der Benutzeroberfläche; er umfasst die Steuerelemente (*Control*) und ermöglich dem Nutzer das Einfügen und Löschen der Steuerelemente. Ein Steuerelement beschreit eine Android View, die durch Touch-Ereignisse Signal and dem Raspberry Pi über die Funkschnittstelle sendet. Eine weitere Aufgabe dieser Schnittstelle ist die Übertragen von Raspberry Pi Nachrichten an der Benutzeroberfläche. Die Raspberry Pi Seite besteht aus folgenden Module: seine Kommunikation Schnittstelle und die API. Die Kommunikation Schnittstelle der Raspberry Pi Sorgt um den Empfang und Weiterleiten vom Signale an die API. Weiterhin überträgt auch die daraus resultierenden Nachrichten an der Android Anwendung. Die API der Benutzer Baustelle; hier kann der Nutzer die Übertragenen Signale beeinflussen.

- **Universalität**

Im Kontext dieser Bachelorarbeit bezeichnet die Universalität, die Möglichkeit die App für unterschiedliche Anwendungsszenarien einzusetzen. Dadurch dass die die Applikation einfach Von Nutzer definierte Signale überträgt und der ihm die Möglichkeit bietet, durch die API diese Signale nach seinem Wunsch zu beeinflussen oder zu implementieren, soll die Universalität des Bedienungssystems sichergestellt.

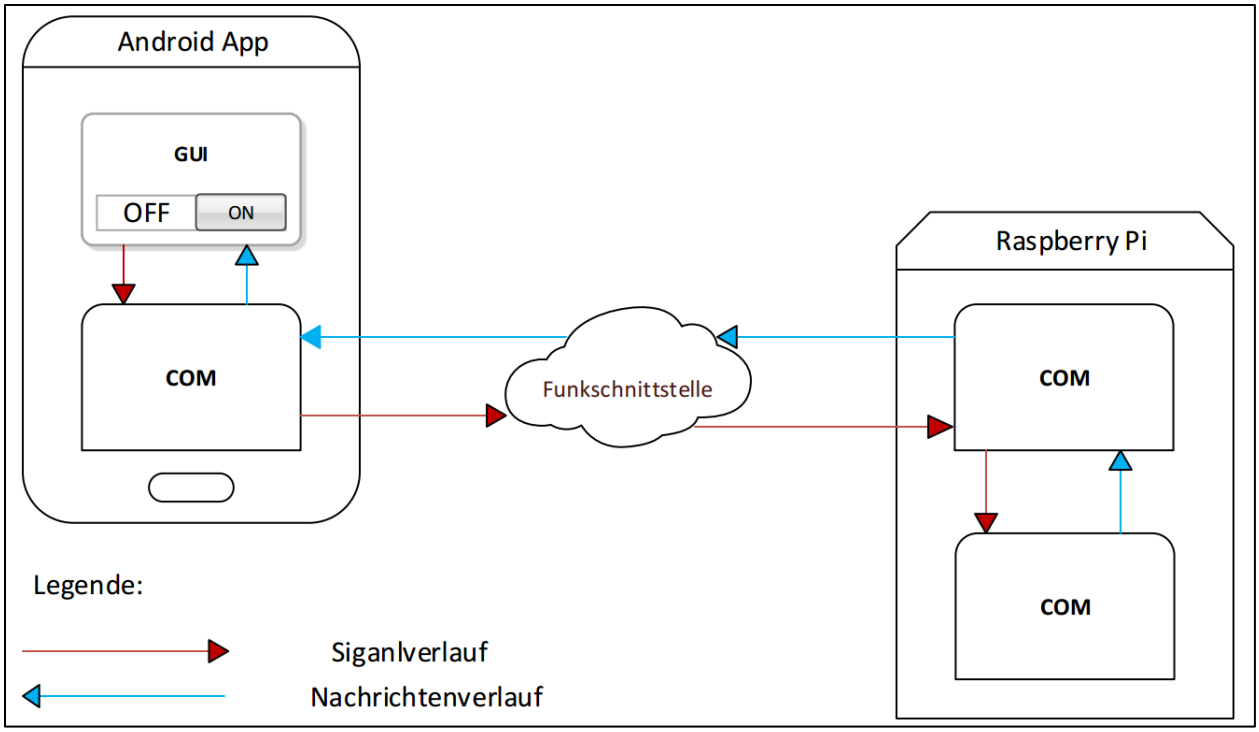

Abbildung 14: Aufbau des Bedienssytems

3.2. Entscheidungen über Implementierung

In den Grundlagen haben wir das Android. Wir haben sowohl bei dem Android Smartphone als auch bei dem Raspberry Pi erfahren welche Funkschnittstelle sie anbietet, welche Programmiersprachen sie Unterstützen und welche Tools Sie haben. In diesem Kapitel werden wir begründetet Entscheidungen für die Implementierung treffen.

3.2.1. Übertragung

Das Thema Übertragung stellt zwei Fragen vor: Die Schnittstelle und die Übertragungsart.

- **Schnittstelle**

Android Smartphone und Raspberry Pi haben gemeinsam Bluetooth und WLAN. Die Übertragung könnte also theoretisch über eins von den Beiden geschehen. Jedoch besitzen Sie Vor- und Nachteile.

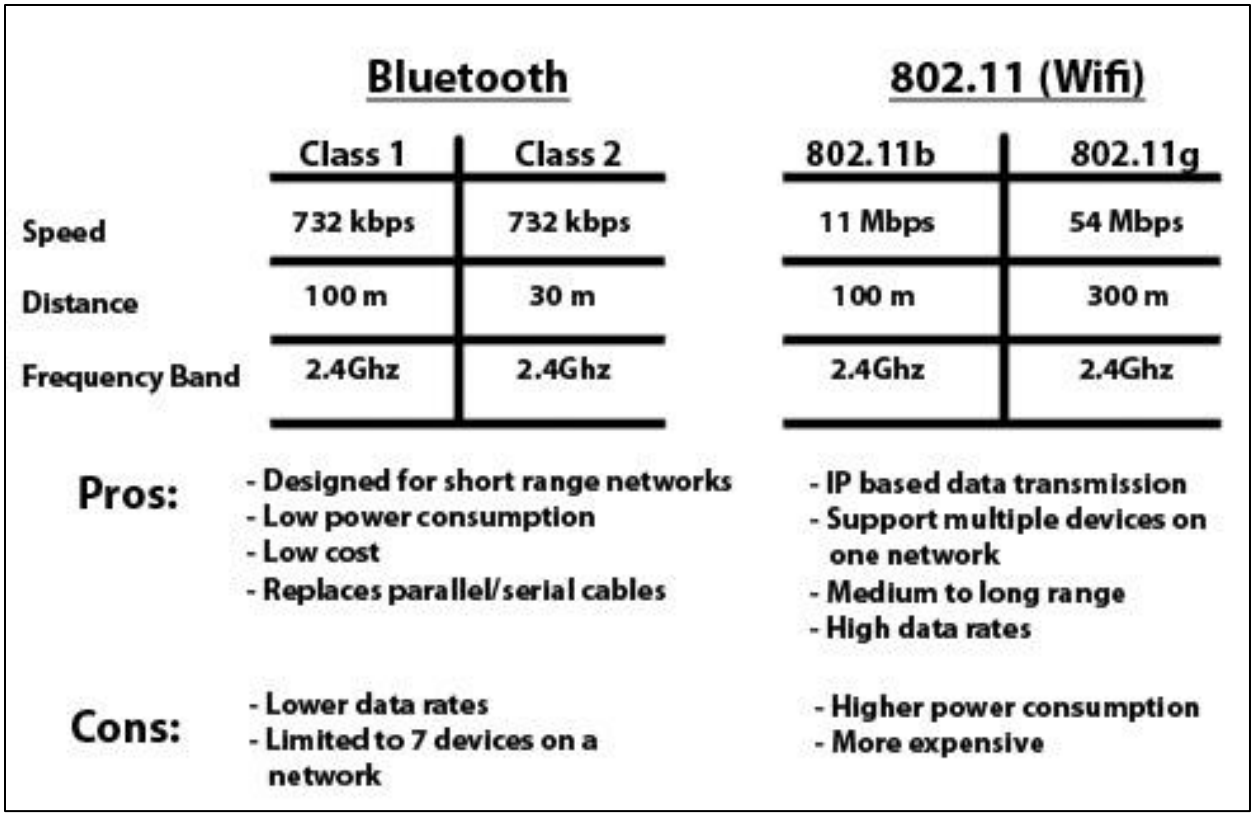

Abbildung 15: Unterschiede, Vor- und Nachteile zwischen Bluetooth und WLAN (20)

die Entscheidung für eins davon sollte vom Systemvoraussetzung und Nutzer Herausforderungen bestimmt werden.

Für diese Arbeit wurde sich für WLAN entschieden. Die wesentliche gründen dafür sind seine breitere Reichweite und die Unterstützung mehrerer Geräte. Es ist bei Eingebettet System besonders wichtig die Maximale Entfernung zwischen Bediener und das Bediente System Groß zu halten. Die in Abbildung 15 erwähnte Nachteile der WLAN treffen in unserem Kontext nicht zu, weil sowohl das Smartphone als auch der Raspberry Pi eingebautes WLAN besitzen.

- **Übertragungsart**

Die WLAN Schnittstelle bietet wiederum zwei übertragungsarten: Wi-Fi Direct WIFI Access Point.

Wi-Fi Direct ist ein Standard zur Datenübermittlung zwischen zwei WLAN-Endgeräten ohne zentralen Wireless Access Point (21). Jedoch unterstützen nicht alle Android Smartphone dieser Technologie und laut (22) Unterstützt der in Raspberry Pi Modell 3 B eingebautes WLAN dieser Technologie nicht. Man kann sich aber ein extra WLAN Modul besorgen.

Ein Access Point ist ein Gerät, das drahtloses lokales Netzwerk erstellt, worauf sich andere Geräte (Client) verbinden können. Hauptziel ist meistens Client der Internetzugang zu ermöglichen. Client können sie sich auch Nachrichten austauschen. In unseren Fall ist der Raspberry Pi der Access Point und das Smartphone der Client. Der eingebautes WLAN Modul der Raspberry Pi kann als Access Point konfiguriert werden und jedes Smartphone Unterstützt den Client-Mode.

Nachteil dieser Übertragungsart ist, dass der Client keines Internetzugangs hat, wenn der Access Point keiner hat.

Bei dieser Bachelorarbeit wurde die Access Point Variante gewählt. Diese Übertragungsart hat den Vorteil, dass mit vielen Geräten Kompatible ist. Ein weiterer Vorteil ist, dass mehrere Geräte sich Gleichzeitig verbinden können. Dieser Vorteil wäre besonders wichtig falls das Bedienungssystem für die Hausautomatisierung verwendet würde.

3.2.2. Raspberry Pi Module

Dieser Abschnitt umfasst die Entscheidung über Komponenten, die den Raspberry Pi umfassen. Es wird erläutern wie Signale empfangen werden sollen. Weiterhin werden Entscheidungen über die Programmiersprachen der Module getroffen.

- **Webserver**

Wie Oben erläutern, ist Hauptziel ein Access Point die Erstellung eines lokalen Netzes und Somit die Verteilung von Internetzugang. Es kann auch für andere Zwecke verwendet werden dafür braucht man aber der richtigen Software. Damit der Access Point Signale vom Client bearbeitet kann, muss er mit einem Webserver ausgestattet sein. Für diese Arbeit wurde sich für Apache Web Server[5] entschieden. Die Signale werden dann von Android Anwendung als HTTP Anfrage gesendet. HTTP Anfragen ermöglichen generell den Client Informationen zum Webserver zu senden, um Formular abzusenden, Datei hochzuladen oder Dateien abzurufen (23). In unseren Fall werden wir Kommunikation Schnittstelle der Raspberry Pi mit Parametern aufrufen.

```
http://www.webserver-adress.com/file_to_call.cgi?param1=val1+param2=val2+param3=val3
```

Abbildung 16: Beispiel HTTP Anfrage mit Parametern

- **Module**

„File_to_call.cgi“ auf Abbildung 16 ist die Schnittstelle auf dem Server, die die Anfrage Empfängt. Es handelt sich um ein CGI-Skript. Das Common Gateway Interface (CGI) ist ein Standard zum Schreiben von Programmen, die über einen Webserver mit einem Client interagieren können, auf dem ein Webbrowser ausgeführt wird. Grundsätzlich interpretiert, verarbeitet er die an ihn gesendeten Informationen und generiert die Antwort, die an den Client zurückgesendet wird. Meistens werden diese Informationen über Umgebungsvariable[6] in das CGI-Skript eingegeben (24).

Wie in den Grundlagen erläutert, kann der Raspberry Pi Python, C/C++ und Java Programmiert werden. Außer C sind anderen Sprachen sogenannten Höhere Programmiersprachen. Der

[5] https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md [10.02.2018]
[6] https://de.wikipedia.org/wiki/Umgebungsvariable [13.02.2018]

Quellcode dieser Programmiersprachen wird erst in Byte-Code umgewandelt der dann in einer Virtuelle Maschine ausgeführt wird. Dadurch ergibt sich eine längere Laufzeit.

Auf (24) wurde dieser Unterschied bewiesen. Dafür wurde ein Programm für dauerhaft Ein-/Ausschalten von LED in Python und C geschrieben und auf dem Raspberry Pi ausgeführt.

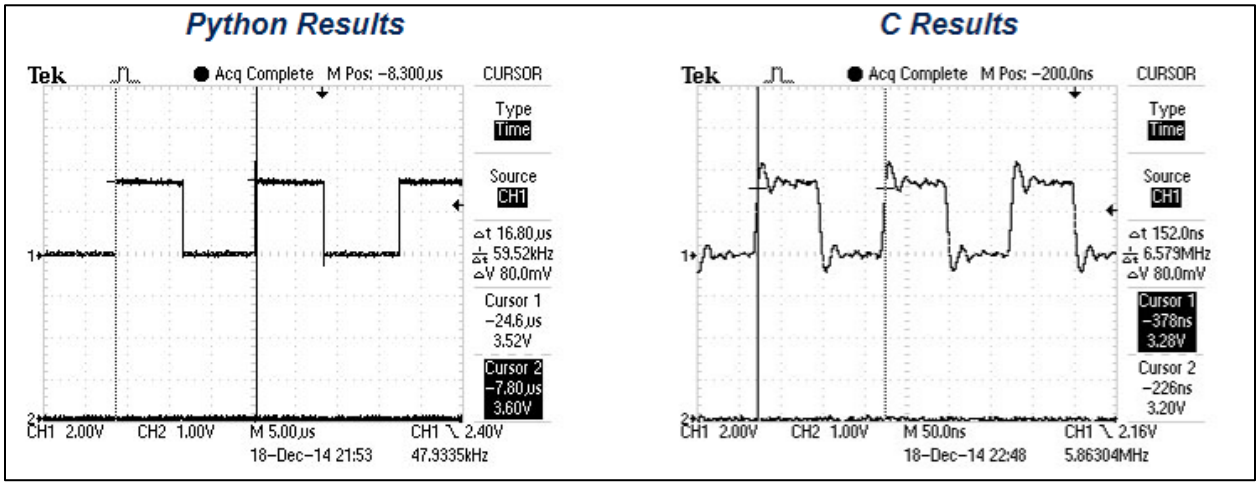

Abbildung 17: Leistung Python und C (24)

Die Zeitachse ist in Mikrosekunde und Nanosekunde bei Python beziehungsweise C. letztgenannt ist also Ca. 1000-mal schneller als Python. Jedoch hat er der Nachteil, dass er Komplexer ist.

Jedoch Reicht die Geschwindigkeit von Höhere Sprache für unseren Anwendungsfall vollkommen aus. Deswegen Wurde die Kommunikation Schnittstelle und die API beziehungsweise in Java und Python Implementiert. Ein Grund Dafür beide Module in verschiedene Sprachen zu implementieren, ist die Modularität des Systems hervorzuheben.

3.2.3. Android Anwendung

Hier ist der Auswahl begrenz; Android Anwendungen können nur in Java programmiert werden. Die Kommunikation Schnittstelle wird dann in einem separaten Paket abgelegt, um seinen Austausch zu erleichtern.

4. Implementierung

Dieses Kapitel umfasst den Praktischen Teil der Bachelorarbeit. Das erste Unterkapitel „Pflichtenheft“ gliedert sich in drei Kategorien „Muss“, „Wunsch“ und „Abgrenzung“ auf. In Stichpunkten wird aufgezählt, welche notwendigen Schritte in der Realisierung vorhanden sein müssen, welche Wunschkriterien eventuell realisiert werden, aber nicht zwingend erforderlich sind und explizite Abgrenzungen, die nicht umgesetzt werden.

Im Anschluss daran folgt die Implementierung von auf Abbildung 14 definierten Module. Schrittweise wird die Implementierung der Bibliothek und der API mithilfe von Bilder und Graphen beschrieben. Jedoch wird nur die wesentlichen Aspekte betrachtet. Danach folgt die

Implementierung der Android Anwendung. Für bessere Verständnis wird die Implementierung mithilfe von Klassendiagramm erklärt. Die Wichtigsten Klassen, Interface und Komponenten werden dargestellt und deren Zusammenhangt verdeutlich.

4.1. Pflichtenheft

Kriterien

-

Ausblick

- Log auf Android anzeigen
- BaseControl class implementieren
- ControlRowView in BaseControl umfassen
- Implementierung von blinkControl
- Mit sigterm pwm prozess beeinflussen ohne es zu beenden und neuzustarten
- App Logik die entsprechen RPI-COm Nachricht Kategorie reagiert.
- Optimiert *get*() in API; param in ein Named Value speichern statt jedes mal die argument durchzusuchen
- AppLib: port beim schliessen der Anwendung freigeben und nicht bein jeder ausmachen
- *HandleBlinkControl*() optimieren
- Ähnlichkeiten zwischen Button und switchcontrol nutzen um code zu optimieren
- Echtzeit ausgabe auf app anstatt warten das alles fertig ist (Beim Blinken z.b.)

4.2. Raspberry Pi Anwendung

4.2.1. API und Seine Bibliothek

<Der Begriff API stammt aus dem englischen Sprachraum und ist die Kurzform von „Application-Programming-Interface“. Frei ins Deutsche Übersetzt bedeutet dies so viel wie „Schnittstelle zur Anwendungsprogrammierung“.>?

In unseren Kontext ist die API die Schnittstelle zur Verwaltung übertragene Signale. Somit ermöglich sie die gewünschte Universalität. Sie ist in der Programmiersprache Python geschrieben. Die Signal Informationen werden beim Aufruf des Skriptes als Argumente übergeben. Die Übergabe geschieht in Form von Name-Wert-Paare (vgl. Abbildung 18).

```
PiControlAPI.py “pin_number=5" "state=0" "mode=11" "interval=0" "number_of_cycles=5" "signal_type=0"
```

Abbildung 18: Beispiel Aufruf API mit Argumenten

Der Zugriff auf diese Argumente in Skripte wird durch die Funktion *sys.argv* ermöglich. Es handelt sich um eine Python Methode, die an einem Skript übergebene Parametern auflistet (25).

Die Bibliothek ist auch in Python geschrieben und ihre Aufgabe ist die die Manipulation der Parameter zu vereinheitlichen und zu erleichtern. Weiterhin bietet Sie grundlegende Methoden zur Verwaltung GPIO-Ports. Sie umfasst unter anderen Methoden zur Ein-/Ausschalten, Blinken, und PWM-Ausgabe der Ports. Sie ist in der API einzubinden/eingebunden.

Methoden zur Verwaltung übertragene Signale

Die wesentlichen Methoden dieser Kategorie sind die *get*-, und *log*-Methoden:

- Die *get*-Methode durchsucht die Argumentliste und gibt den werden der Übergebene Argumentname zurück. Der Methode kann Eins bis Drei Parametern übergeben werden:
  - *key*: ist eine Zeichenkette (String[7]), die den Argument-Name beschreibt. Er muss immer übergeben werden.
  - *isRequired*: vom Typ *boolean* ist *False* wenn vom Nutzer nicht übergeben. Er Bestimmt, ob das Argument erforderlich ist, wenn *True,* beendet sich das Programm mit Fehler Ausgabe auf die Konsole, falls der Wert nicht gefunden wurde.
  - *isString*: ist auch vom Typ *boolean* und seines Wert ist Standartmäßig *False*. Der Parameter legt fest ob der Wert als String behandelt werden soll oder nicht; beim *False* wird er als Integer behandelt.
- Die *log*-Methode wurde für eine strukturierte und einheitliche Konsole Ausgabe implementiert. Die Ausgabe mit dieser Methode beinhaltet das Datum, die Uhrzeit an denen die Ausgabe generiert wurde und die Nachricht selber. Sie werden in Form von HTML Element (*span*) angezeigt und mittels der HTML Klassen-Attribut unter verschiedene Kategorien zugeordnet. Zwecks ist die Android Anwendung zu ermöglichen, die Nachtricht-Kategorie zu erkennen und entsprechend darauf zu reagieren. Beim Aufruf der Funktion muss zusätzlich zur Nachricht auch die Abkürzung der Nachricht-Kategorie (als Character[8]) übergeben werden. Es existiert schon folgende Kategorien (mit Abkürzungen):
  - *ERROR* (*e*) zum Ausgeben einem Fehler
  - *WARNING* (*w*) zum Ausgeben einer Warnung
  - *TODO* (*t*) zum Ausgeben einen noch nicht implementierte feature
  - *INFO* (*i*) zum Ausgeben von Standard Nachrichten

Methoden Zur Verwaltung GPIO-Ports

Außer *handlePwmControl*-Methode handelt es sich um Wrapper Methoden, die Ausgabe von Signale auf GPIO-Port vereinfachen indem sie eine Reihe von Methoden der RPi.GPIO[9] Bibliothek einkapselt, die für die Ausgabe ein bestimmtes Signal notwendig sind. Die RPi.GPIO Bibliothek

[7] https://en.wikipedia.org/wiki/String_(computer_science) [12.02.18]
[8] https://en.wikipedia.org/wiki/Character_(computing) [12.02.18]
[9] https://pypi.python.org/pypi/RPi.GPIO [12.02.18]

wurde in App Bibliothek eingebunden. Zu diesen Methoden zählen *handleSwitchControl*, *handleButtonControl*, *handeBlinkControl*, und *handlePwmControl*. Jede Methode wurde aufgebaut um das Signal eine bestimmte Control zu behandelt. *handleBlinkControl* behandelt beispielweise Signale vom *BlinkControl*.

```
def handleBlinkControl():
        GPIO.setmode(get(KEY_MODE, True))
        pin = get(KEY_PIN_NUMBER, True)
        GPIO.setup(pin, GPIO.OUT)
        interval = get(KEY_INTERVAL)
        nrOfCycles = get(KEY_NUMBER_OF_CYCLES, True)
        if(nrOfCycles == 0):
                try:
                        while(True):
                                GPIO.output(pin, GPIO.HIGH)
                                time.sleep(interval)
                                GPIO.output(pin, GPIO.LOW)
                                time.sleep(interval)
                except KeyboardInterrupt:
                        GPIO.output(pin, GPIO.LOW)
                        GPIO.cleanup()
                        log("d", "Programm Interrupted with CTRL + C")
        else:
                try:
                        counter = 0
                        while(counter < nrOfCycles):
                                GPIO.output(pin, GPIO.HIGH)
                                time.sleep(interval)
                                GPIO.output(pin, GPIO.LOW)
                                time.sleep(interval)
                                counter += 1
                except KeyboardInterrupt:
                        GPIO.output(pin, GPIO.LOW)
                        GPIO.cleanup()
                        log("d", "Programm Interrupted with CTRL + C")
```

Abbildung 19: Methode *handleBlinkControl* der App Bibliothek

Bevor dieser Methode aufgerufen wird soll erst geprüft werden, ob der Signal vom Typ *BlinkControl* ist. Dies geschieht mithilfe der *getName*-Methode.

```
63          elif(AppLib.getName() == AppLib.BLINK_CONTROL_NAME):
64                  AppLib.handleBlinkControl()
65          else:
66                  AppLib.log("t", "Control Handler not implemented yet...")
67
```

Abbildung 20: Beispiel Signal-Typ Überprüfung

Außer *hanlePwmControl* Funktionieren andere Methoden nach demselben Schema.

Wie eingangs erwähnt ist die *handlePwmControl*-Methode anders implementiert. Die Methode ist zur Behandlung PWM Signale. Der Raspberry Pi unterstützt keine Hardware PWM, daher wird er mit einer Software-Schleife emuliert. In einem getrennten Thread wird das Signal dauerhaft gesetzt und zurückgesetzt. Beim beenden des Programms wird das Thread auch beendet und somit das PWM ausgeschaltet (26).

Eine Schleife wäre aber in unseren Kontext ungünstig, weil die Anwendung würde ja kein Signal mehr empfangen bis die Schleife beendet wird. Damit unseren PWM an bleibt wurde mithilfe dieses Beispiels[10] ein Daemon[11] implementiert. Gemäß dem Beispiel wurde das Daemon in eine andere Datei ausgelagert. Das Daemon Paket muss erst installiert werden.

```
#!/usr/bin/python
import sys
import RPi.GPIO as GPIO
import time
from daemon import runner

params = sys.argv

class PwmDaemon():
    def __init__(self):
                self.stdin_path = '/dev/null'
                self.stdout_path = '/dev/null'
                self.stderr_path = '/dev/null'
                self.pidfile_path =  '/tmp/foo.pid'
                self.pidfile_timeout = 5
                self.ch = int(params[2])
                self.freq = int(params[3])
                self.dc = int(params[4])
    def run(self):
        #while True:
            #handlePwmOutput(self.ch, self.freq, self.dc)
                GPIO.setmode(GPIO.BCM)
                GPIO.setup(self.ch, GPIO.OUT)
                p = GPIO.PWM(self.ch, self.freq)
                p.ChangeDutyCycle(self.dc)
                p.start(self.dc)
                #log("t", "Look for a nicer Solution for the PWM")
                #log("d", "outputing PWM... Press CRTL + C to Stop")
                try:
                        while(True):
                                pass
                except KeyboardInterrupt:
                        p.stop()

pwmDaemon = PwmDaemon()
daemon_runner = runner.DaemonRunner(pwmDaemon)
daemon_runner.do_action()
```

Abbildung 21: PWM Daemon Skript

Die Datei muss beim Aufruf in der Reihenfolge der Port Nummer, die Frequenz und der Duty Cycle der PWM übergeben werden. Der Aufruf kann von ein Python Skript mithilfe der Python Methode *os.system* ausgeführt werden.

4.2.2. Kommunikation Schnittstelle

Die Aufgabe dieses Modul ist die Verarbeitung von Signale und die Bereitstellung der Konsole Ausgabe der API. Es handelt sich Grundsätzlich um eine Java Anwendung die zusätzlich zur Verarbeitung und Weiterleitung von Signale an den API auch eine HTML Seite generiert, die

[10] https://stackoverflow.com/a/9047339/7614256 [13.02.2018]
[11] https://pypi.python.org/pypi/python-daemon/ [13.02.18]

Informationen über das verarbeitete Signal und die Konsole Ausgabe der API in einer bestimmten Form beinhaltet. Die HTML Seite kann dann in Browser angeschaut werden oder wie in unseren Fall durch eine weitere Anwendung (Android Applikation) verarbeite. Die Anwendung besteht aus eine Java-klasse, deren wichtigste Methoden *buildCmd* und *callApi* sind. *buildCmd* generiert das Kommando, die zum Aufrufen des API Notwendig sind. Das Kommando besteht unter anderen aus der API Umgebung (Python – API ist in Python), sein Name, sein Absoluter Path, und den Parametern. *callApi* ruft API mittels generierte Kommando auf und fügt die vom API generierte Ausgabe in der HTML-Seite ein.

Here are the CGI environment variables/value pairs passed in from the CGI script:

CONTENT_TYPE
::
CONTENT_LENGTH
::
REQUEST_METHOD
:GET:
QUERY_STRING
:state=0&pin_number=5&short_description=aviDesc&direction=0&signal_type=0&mode=11&name=SwitchControl:
SERVER_NAME
:192.168.42.1:
SERVER_PORT
:80:
SCRIPT_NAME
:/cgi-bin/RpiComIface.cgi:
PATH_INFO
::

Here is the API Call:

callInterpreter(python /usr/lib/cgi-bin/RpiInterpreter.py state=0 pin_number=5 short_description=aviDesc direction=0 signal_type=0 mode=11 name=SwitchControl)

Here is the API output:

2018-02-13 11:59:55 __main__
2018-02-13 11:59:55 get(name)
2018-02-13 11:59:55 key: 'name' - value: 'SwitchControl'
2018-02-13 11:59:55 get(mode)
2018-02-13 11:59:55 key: 'mode' - value: '11'
2018-02-13 11:59:55 get(pin_number)
2018-02-13 11:59:55 key: 'pin_number' - value: '5'
2018-02-13 11:59:55 get(state)
2018-02-13 11:59:55 key: 'state' - value: '0'
2018-02-13 11:59:55 Port 5 cleaned

Abbildung 22: Vom COM Schnittstelle der Raspberry Pi generierte HTML Seite nach einem Signaldurchlauf

Die Anwendung wird vom CGI-Skript über Umgebungsvariable mit Signal Informationen versorgt und ausgeführt. Damit der Zugriff und die Verarbeitung von Signal Informationen leichter werden, wurde die Hier[12] implementierte Bibliothek verwendet.

[12] https://www.javaworld.com/article/2076863/java-web-development/write-cgi-programs-in-java.html [13.02.18]