

Project PI



Project PI Machbarkeitsstudie

Version 6.0
20.04.2015

Autor:
Johannes Ucel, Yehezkel Sivan, Michael Stöger, Antonio Pavic

QS:
Johannes Ucel, Michael Stöger

Status:
Final

<i>Version</i>	<i>Autor</i>	<i>QS</i>	<i>Datum</i>	<i>Status</i>	<i>Kommentar</i>
1.0	Johannes Ucel	Michael Stöger	24.02.2015	Draft	Einführung - Produktfunktionen
2.0	Michael Stöger	Johannes Ucel	05.03.2015	Draft	Produktfunktionen – techn. Machbarkeit
3.0	Antonio Pavic	Johannes Ucel	05.03.2015	Draft	Wirtsch. Machbarkeit + Nutzwertanalyse
4.0	Yehezkel Sivan	Johannes Ucel	06.03.2015	Draft	Projektorganisation – Management Summary
5.0	Johannes Ucel	Michael Stöger	09.03.2015	Draft	Korrektur der früheren Versionen
6.0	Johannes Ucel	Michael Stöger	20.04.2015	Final	Hinzufügen der Tests

Inhaltsverzeichnis

1 Einführung	5
2 Projektdaten	5
2.1 Projektbeschreibung	5
3 Voruntersuchung des Produkts	6
3.1 Ist-Zustand.....	6
3.2 Alternativen am Markt.....	6
4 Produktauswahl	6
4.1 Trendanalyse	6
4.2 Marktanalyse	6
5 Sollzustand	7
5.1 Muss – Ziele	7
5.1.1 Interaktives Steuern.....	7
5.1.2 Web - Blog.....	7
5.1.3 Benutzerverwaltungssystem	7
5.1.4 Zuverlässigkeit und Effizienz	7
5.1.5 Erweiterbarkeit.....	7
5.1.6 Updates und Verbesserungen.....	7
5.2 Kann – Ziele	8
5.2.1 Kompatibilität mit anderen Systemen.....	8
5.2.2 Spracheingabe	8
6 Produktfunktionen	9
6.1 Systemsteuerung.....	9
6.2 Sensitive Elemente	16
6.3 Datenübertragung	23
6.4 Steuerung über App	26

7 Aktivitätsdiagramm	28
8 Technische Machbarkeit	29
8.1 Technologien.....	29
8.2 Umsetzung.....	29
8.3 Sensoren – Tests	30
8.3.1 Taster.....	30
8.3.2 Kombi LED	30
8.3.3 Annäherungssensor	31
8.3.4 Helligkeitssensor	31
8.3.5 WLAN – Kommunikation	32
9 Wirtschaftliche Machbarkeit	33
9.1 Personalaufwand.....	33
9.2 Investitionsaufwand	33
9.3 Nutzen.....	33
9.4 Risikoanalyse	34
9.4.1 Personenausfall	34
9.4.2 Zeitliche Risiken	34
9.4.3 Technische Risiken	34
10 Nutzwertanalyse	35
11 Projektorganisation	36
12 Projektplanung	37
13 Management Summary	38

1 Einführung

In dem Projekt „Project PI“, geht es um ein **Plüschtier** (in unserem Fall ein „Pi-kachu“), welches mit verschiedenen **technischen Funktionen** ausgestattet wird. Dieses Plüschtier soll speziell Kindern Spaß machen, da man zusätzlich zum Spielen auch aufgrund technischer Funktionen **Neugier zur Technik** entwickelt.

Unterstützt werden die technischen Funktionen des Plüschtiers von einer **Smartphone – App** (Android), wo man die Möglichkeit hat, mit dem Plüschtier andere Funktionen auszuprobieren.

Weiteres hat jeder die Möglichkeit, die Projektfortschritte in einem speziell für das Projekt kreierten **Web – Blog** mitzuverfolgen. Weiteres findet man dort auch **Tutorials** und **Hilfestellungen**, falls jemand ein ähnliches Projekt verfolgt.

2 Projektdaten

2.1 Projektbeschreibung

Im Rahmen des Projektes „Project PI“ wird sowohl eine **Software**, welche auf einem Raspberry Pi laufen wird, als auch eine **Smartphone – App** (Android), welche zur **erweiterten Steuerung** dient, entwickelt.

So haben Benutzer die Möglichkeit, mittels der App **Befehle an das Plüschtier** (bzw. an den Raspberry Pi) zu geben.

Weiteres kann die **Funktionalität**, welche natürlich auch Hardwareabhängig ist, immer **ergänzt** und den Wünschen nach **angepasst** werden.

3 Voruntersuchung des Produkts

3.1 Ist-Zustand

Aktuell gibt es Spielzeuge, welche ähnliche Funktionen aufweisen, jedoch nicht in Form eines Raspberry's. Diese Spielzeuge verwenden andere Technologien, welche jedoch nicht die Vorteile des Raspberry's aufweisen, welche oben genannt wurden.

3.2 Alternativen am Markt

Zurzeit sind keine ähnlichen Projekte bekannt, wobei man aber beachten muss, dass die „Raspberry – Community“ sehr groß ist und nicht jeder sein Produkt vermarktet bzw. veröffentlicht. Prinzipiell hat jeder die Möglichkeit, einen Raspberry Pi in einem Plüschtier einzubauen und zu programmieren.

Durch diese „Einmaligkeit“ lässt sich das fertige Produkt sehr gut vermarkten, wodurch höchstwahrscheinlich auch ein neuer Trend entstehen wird.

4 Produktauswahl

4.1 Trendanalyse

Der Trend geht dahin, dass sich die Technik in alle Bereich weiterentwickelt und auch die Spielzeug – Branche nach moderneren und technisch versierteren Spielzeugen strebt.

4.2 Marktanalyse

Das finale Produkt des Projektes „Project Pi“ erfüllt die Marktanforderungen und ist zurzeit eines der wenigen „Gesamtsysteme“ im Bereich der Spielzeug – Industrie. Das System wirbt mit:

- einer einfachen und umfangreichen Bedienung
- flexibler Anpassungsfähig
- laufende Updates und Verbesserungen
- einem umfangreichen und qualifiziertem Support
- spielerisches Erforschen der Technik
- für den internationalen Markt geeignet (**multilingual**)

5 Sollzustand

5.1 Muss – Ziele

5.1.1 *Interaktives Steuern*

Jeder Benutzer soll die Möglichkeit haben, das Plüschtier sowohl direkt mit Tastern bzw. Sensoren, als auch mittels der Smartphone – App, zu bedienen.

Auf jede Betätigung soll das Plüschtier mit der vorprogrammierten Aktion reagieren.

So kann man mit dem Soundboard der Smartphone – App Sounds von dem Plüschtier ausgeben lassen.

5.1.2 *Web - Blog*

Jeder hat die Möglichkeit, das Projekt mitzuverfolgen bzw. die aktuellen Fortschritte zu beobachten. Abschließend soll dieser Web – Blog am Ende als „Tutorial“ dienen, wodurch interessierte Hobby – Bastler ein ähnliches Modell entwickeln können.

5.1.3 *Benutzerverwaltungssystem*

Innerhalb des Unternehmens gibt es ein Benutzerverwaltungssystem, wo Mitarbeiter abhängig von ihrer Position Berechtigungen erhalten. (z.B. Buchhaltung hat Zugriff auf Finanzen)

5.1.4 *Zuverlässigkeit und Effizienz*

Sowohl beim direkten Benutzen des Plüschtiers, als auch bei der Verwendung der Smartphone – App soll eine Fehlerrate von kleiner als 0,1% vorgewiesen werden. Um einen seriösen Eindruck zu vermitteln, sollten die Ladezeiten der App bzw. des Web – Blogs möglichst kurz und konstant gehalten sein (Dementsprechend größere Server).

5.1.5 *Erweiterbarkeit*

Das System soll jederzeit von seinen Funktionen erweiterbar sein (Sei es nun programmiertechnisch oder hardwaremäßig). Weiteres sollen die Systemsprachen anpassbar sein, wobei standardmäßig die Sprachen Deutsch und Englisch verfügbar sind und weitere Sprachen bei Bedarf hinzugefügt werden können.

5.1.6 *Updates und Verbesserungen*

Es wird laufend an Verbesserungen gearbeitet, sodass die Software immer am aktuellen Stand ist. Abhängig von zukünftigen Python – Version lassen sich gewisse Vorgänge in der Zukunft einfacher bzw. schnell durchführen. Weiteres wird darauf geachtet, dass die App auch mit den neueren Android-Versionen funktioniert.

5.2 Kann – Ziele

5.2.1 *Kompatibilität mit anderen Systemen*

Ein Benutzer soll die Möglichkeit haben, mit den gängigen Smartphone - Betriebssystemen (Windows Phone, Android, iOS, Ubuntu Phone) die App zu nutzen.

5.2.2 *Spracheingabe*

Ein Benutzer soll die Möglichkeit haben, dem Plüschtier sprachlich Befehle zu geben. Hier muss jedoch sowohl eine funktionsfähige Spracherkennungs – Engine, als auch ein Mikrofon, welches nicht leicht durch Nebengeräusche bzw. Erschütterungen beeinflusst wird, vorhanden sein.

6 Produktfunktionen

6.1 Systemsteuerung

Der Raspberry Pi muss ohne Tastatureingaben gesteuert werden können.

/LF10/ Ausfallsicherheit

Die Software zur Steuerung des Kuscheltiers muss immer laufen, bis sie von Benutzer absichtlich beendet wird. Dies umfasst: Systemstart, Programmabsturz

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Hoch	Niedrig	MH
Name	Ausfallsicherheit /LF10/			
Art	Hintergrundprozess			
Kurzbeschreibung	Die Software des Kuscheltiers muss laufen, solange dies vom Benutzer gewünscht ist.			
Auslöser	Systemstart, Programmcrash			
Ergebnis	Programm läuft			
Akteure	Benutzer, Software			
Eingehende Informationen	Zustand der Software			
Vorbedingungen	System muss gestartet sein/Software muss laufen			
Nachbedingungen	Software muss gestartet sein			

/LF20/ Remote beenden

Die Software kann über Fernsteuerung per Android App beendet werden.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Mittel	Mittel	SH
Name	Remote beenden /LF20/			
Art	Anwendungsfall			
Kurzbeschreibung	Android App kann Software beenden.			
Auslöser	Benutzer beendet Software mit Android App			
Ergebnis	Software ist beendet			
Akteure	Benutzer			
Eingehende Informationen	Zeitpunkt zum Beenden			
Vorbedingungen	Software muss laufen, Android Client ist mit Software verbunden			
Nachbedingungen	Software muss beendet sein			

/LF30/ Remote neustart

Die Software kann über die Android App neu gestartet werden.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Mittel	Mittel	SH
Name	Remote neustart /LF30/			
Art	Anwendungsfall			
Kurzbeschreibung	Software kann über App neu gestartet werden			
Auslöser	Benutzer startet Software über App neu			
Ergebnis	Software ist neu gestartet			
Akteure	Benutzer			
Eingehende Informationen	Zeitpunkt zum neu Starten			
Vorbedingungen	Software muss laufen, Android Client muss mit Software verbunden sein			
Nachbedingungen	Software muss neu gestartet sein			

/LF40/ Remote herunterfahren

Der Raspberry kann über die Android App heruntergefahren werden.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice Have to Have
UseCase		Hoch	Mittel	MH
Name	Remote herunterfahren /LF40/			
Art	Anwendungsfall			
Kurzbeschreibung	Raspberry Pi wird über Android App heruntergefahren.			
Auslöser	Benutzer fährt Raspberry über App herunter			
Ergebnis	Raspberry Pi ist ausgeschaltet			
Akteure	Benutzer			
Eingehende Informationen	Zeitpunkt des Herunterfahrens			
Vorbedingungen	Software muss laufen, Android Client muss mit Software verbunden sein			
Nachbedingungen	Raspberry Pi ist ausgeschaltet			

/LF41/ Alarm bei Überhitzung

Überhitzt der Raspberry Pi, gibt die Software den entsprechenden Warnton in einer Endlosschleife wieder und stoppt die Wiedergabe erst wieder, wenn die Temperatur wieder unter einen bestimmten Punkt gefallen ist.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Hoch	Mittel	MH
Name	Alarm bei Überhitzung /LF41/			
Art	Hintergrundprozess			
Kurzbeschreibung	Software warnt mit Sound vor Überhitzung			
Auslöser	CPU des Raspberry Pi überhitzt			
Ergebnis	Warnton wird wiedergegeben			
Akteure	Software, Raspberry Pi			
Eingehende Informationen	Aktuelle CPU Temperatur			
Vorbedingungen	Software muss laufen			
Nachbedingungen	Sound wird wiedergegeben			

/LF42/ Herunterfahren bei Überhitzung

Erreicht die CPU des Raspberry Pi einen kritischen Punkt schaltet die Software den Raspberry automatisch ab, um Schäden am System zu vermeiden.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Hoch	Mittel	MH
Name	Herunterfahren bei Überhitzung /LF42/			
Art	Hintergrundprozess			
Kurzbeschreibung	Herunterfahren bei überhitzen der CPU			
Auslöser	CPU erreicht kritische Temperatur			
Ergebnis	Raspberry Pi ist ausgeschalten			
Akteure	Software, Raspberry Pi			
Eingehende Informationen	Aktuelle CPU Temperatur			
Vorbedingungen	Software muss laufen			
Nachbedingungen	Raspberry Pi muss ausgeschalten sein			

/LF43/ Steuerung über SSH

Am Raspberry Pi läuft ein SSH Server über den der Raspberry Pi gewartet werden kann. Das umfasst: Softwareupdates, Einstellungen.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Hoch	Niedrig	MH
Name	Steuerung über SSH /LF43/			
Art	Wartung			
Kurzbeschreibung	Raspberry Pi kann über SSH gesteuert werden			
Auslöser	Eingehende SSH Verbindung			
Ergebnis	Raspberry wird über SSH gesteuert			
Akteure	Benutzer			
Eingehende Informationen	Benutzername, Passwort			
Vorbedingungen	Raspberry Pi muss laufen, SSH Server muss laufen			
Nachbedingungen	Raspberry Pi wird über SSH gesteuert			

6.2 Sensitive Elemente

Das Verhalten der Software soll größtenteils durch Sensoren wie Taster, Annäherungssensor und Lichtsensor bestimmt werden. Die Software kann über Licht und Ton Rückmeldungen geben.

/LF50/ Selfie Knopf

Wird der Taster auf der rechten Hand gedrückt, wird der Fotomechanismus ausgelöst. Dadurch spielt die Software den spezifischen Ton und schießt dann über die angeschlossene Kamera ein Foto. Dieses Foto wird dann in /Fotos abgelegt.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase Name Selfie Knopf /LF50/ Art Anwendungsfall Kurzbeschreibung Rechter Taster schießt ein Foto Auslöser Benutzer drückt rechten Taster Ergebnis Foto liegt in /Fotos Akteure Benutzer Eingehende Informationen Sensordaten über Taster Vorbedingungen Software muss gestartet sein Nachbedingungen Bild muss in /Fotos gespeichert sein		Mittel	Hoch	MH

/LF60/ Lichtsensor - Hell

Ändert sich die Raumhelligkeit von Dunkel nach Hell, so wird der spezifische Ton abgespielt und die Farbe der Augen vorübergehend auf weiß geändert.

Funktion	Nutzen	Aufwand	Must Have Should Have Nice to Have
<p>UseCase</p> <p>Name Lichtsensor – Hell /LF60/</p> <p>Art Anwendungsfall</p> <p>Kurzbeschreibung Raumhelligkeit ändert auf Hell</p> <p>Auslöser Raumhelligkeit ändert sich</p> <p>Ergebnis Ton wird abgespielt, Augenfarbe geändert</p> <p>Akteure Benutzer, Umgebung</p> <p>Eingehende Informationen Raumhelligkeit</p> <p>Vorbedingungen Software muss laufen</p> <p>Nachbedingungen Ton ist abgespielt, Augenfarbe wieder normal</p>	Mittel	Mittel	MH

/LF70/ Lichtsensor - Dunkel

Ändert sich die Raumhelligkeit von Hell nach Dunkel, wird der spezifische Ton abgespielt und die Augenfarbe vorübergehend auf grün geändert.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase Name Lichtsensor – Dunkel /LF70/ Art Anwendungsfall Kurzbeschreibung Raumhelligkeit ändert sich von Hell nach Dunkel Auslöser Raumhelligkeit ändert von Hell nach Dunkel Ergebnis Spezifischer Ton wird abgespielt, Augenfarbe ändert sich Akteure Benutzer, Umgebung Eingehende Informationen Sensordaten => Helligkeit Vorbedingungen Software muss laufen Nachbedingungen Sound ist abgespielt, Augen wieder in Normalzustand		Mittel	Mittel	MH

/LF80/ Taster Bauch

Wird der Taster am Bauch gedrückt, wird der spezifische Ton abgespielt und die Augenfarbe vorübergehend geändert.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Mittel	Mittel	MH
Name	Taster Bauch /LF80/			
Art	Anwendungsfall			
Kurzbeschreibung	Taster am Bauch wird gedrückt			
Auslöser	Taster am Bauch			
Ergebnis	Spezifischer Ton wird gespielt, Augenfarbe wird geändert			
Akteure	Benutzer			
Eingehende Informationen	Sensordaten => Taster am Bauch			
Vorbedingungen	Software muss laufen			
Nachbedingungen	Ton wurde abgespielt, Augenfarbe wieder normal			

/LF90/ Taster – linke Hand

Wird der Taster auf der linken Hand gedrückt, wird der Donnerblitz Sound gespielt und die Farbe der Augen auf Gelb gesetzt.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase Name Taster – linke Hand /LF90/ Art Anwendungsfall Kurzbeschreibung Taster an der linken Hand wird gedrückt Auslöser Taster => linke Hand Ergebnis Spezifischer Ton wird gespielt, Augenfarbe wird geändert Akteure Benutzer Eingehende Informationen Sensordaten => Taster linke Hand Vorbedingungen Software muss laufen Nachbedingungen Ton wurde abgespielt, Augenfarbe wieder normal		Mittel	Mittel	MH

/LF100/ Taster am Ohr

Wird der Taster am Ohr gedrückt wird der spezifische Ton abgespielt und die Augenfarbe auf blau geändert.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase Name Taster am Ohr /LF100/ Art Anwendungsfall Kurzbeschreibung Taster am Ohr wird gedrückt und löst eine Aktion aus Auslöser Taster am Ohr Ergebnis Spezifischer Ton wird abgespielt, Augenfarbe wird geändert Akteure Benutzer Eingehende Informationen Sensordaten => Taster am Ohr Vorbedingungen Software muss laufen Nachbedingungen Ton ist abgespielt, Augenfarbe wieder normal		Mittel	Mittel	MH

/LF110/ Annäherung

Nähert sich ein Gegenstand dem Kuscheltier an, wird der spezifische Ton gespielt und die Augenfarbe auf türkis geändert.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase Name Annäherung /LF100/ Art Anwendungsfall Kurzbeschreibung Gegenstand nähert sich Annäherungssensor Auslöser Gegenstand wird von Annäherungssensor registriert Ergebnis Spezifischer Ton wird gespielt, Augenfarbe wird geändert Akteure Benutzer Eingehende Informationen Sensordaten => Annäherungssensor Vorbedingungen Software muss laufen Nachbedingungen Ton ist abgespielt, Augenfarbe wieder normal		Mittel	Hoch	MH

6.3 Datenübertragung

Fotos, welche am Gerät aufgenommen wurden, müssen über das Netzwerk an ein anderes Gerät übermittelt werden können.

/LF120/ Samba Server

Am Raspberry Pi läuft ein Samba Server, welcher den Ordner „/Fotos“ für alle ohne Anmeldung im Netzwerk freigibt.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Hoch	Niedrig	SH
Name	Samba Server /LF120/			
Art	Hintergrundprozess			
Kurzbeschreibung	Auf dem Raspberry Pi wird der Ordner /Fotos mit Samba freigegeben			
Auslöser	Raspberry Pi startet			
Ergebnis	Samba Server läuft am Raspberry			
Akteure	System			
Eingehende Informationen	Startzeitpunkt des Systems			
Vorbedingungen	Raspberry Pi muss laufen			
Nachbedingungen	Samba Server läuft			

/LF130/ Fotos in App

Über die Android App können die Fotos in den Speicher des Android Geräts verschoben werden.

Funktion	Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase Name Fotos in App /LF130/ Art Anwendungsfall Kurzbeschreibung Über die Android App können die Fotos in den Telefonspeicher übertragen werden Auslöser Benutzer startet Dateiübertragung via App Ergebnis Fotos sind im Telefonspeicher Akteure Benutzer Eingehende Informationen Zielordner Vorbedingungen Software muss laufen, Android Client mit Software verbunden Nachbedingungen Fotos sind im Telefonspeicher gespeichert	Mittel	Hoch	NH

/LF131/ Darstellung über HDMI

Über den eingebauten HDMI Ausgang des Raspberry Pi wird das letzte Bild, welches mit der Kamera aufgenommen wurde, angezeigt.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Mittel	Mittel	NH
Name	Darstellung über HDMI /LF131/			
Art	Hintergrundprozess			
Kurzbeschreibung	Letztes Bild wird über HDMI ausgegeben			
Auslöser	Bild wird mit Kamera aufgenommen			
Ergebnis	Bild wird am Bildschirm angezeigt			
Akteure	Software			
Eingehende Informationen	Neues Bild			
Vorbedingungen	Software muss laufen, Bild muss vorhanden sein			
Nachbedingungen	Bild ist am Bildschirm sichtbar			

6.4 Steuerung über App

Die Android App soll das System nicht nur herunterfahren und neustarten können, sondern auch Töne ausgeben und die Augenfarbe ändern können.

/LF140/ Augenfarbe bestimmen

Die Augenfarbe des Kuscheltiers soll über die Android App bestimmt werden können. Dazu stehen Buttons mit den möglichen Farben zur Verfügung.

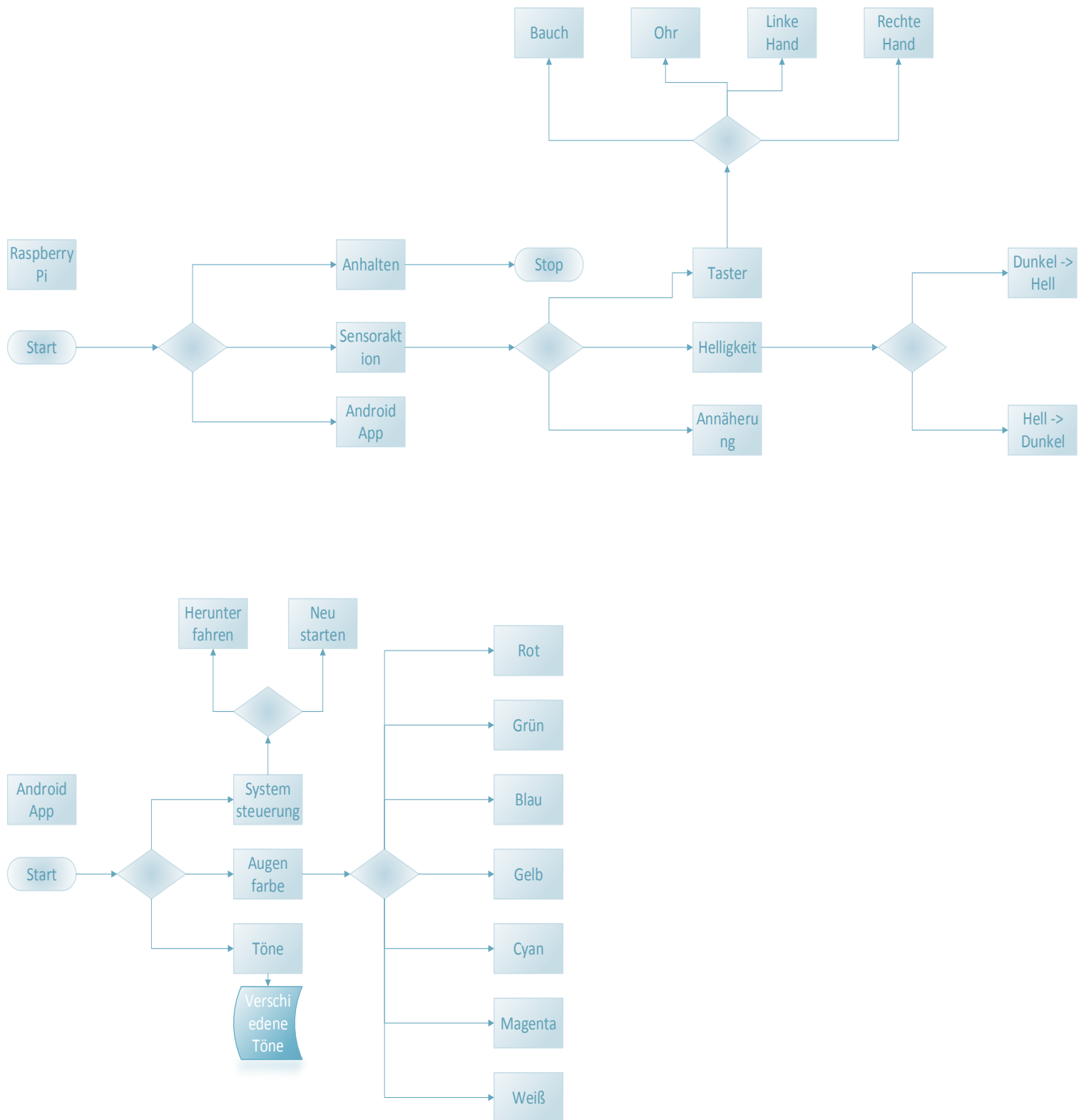
Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase		Mittel	Hoch	SH
Name	Augenfarbe bestimmen /LF140/			
Art	Anwendungsfall			
Kurzbeschreibung	Augenfarbe wird über Android App verändert			
Auslöser	Button in Android App wird betätigt			
Ergebnis	Augenfarbe ändert sich			
Akteure	Beutzer			
Eingehende Informationen	Gewünschte Farbe			
Vorbedingungen	Software muss laufen, Android Client ist mit App verbunden			
Nachbedingungen	Augenfarbe ist geändert			

/LF150/ Töne wiedergeben

Über Buttons in der Android App ist es möglich verschiedene im Kuscheltier gespeicherte Töne wiederzugeben.

Funktion		Nutzen	Aufwand	Must Have Should Have Nice to Have
UseCase Name Töne wiedergeben /LF150/ Art Anwendungsfall Kurzbeschreibung Töne werden über Android App wiedergegeben Auslöser Benutzer betätigt Button in Android App Ergebnis Ton wird wiedergegeben Akteure Benutzer Eingehende Informationen Gewünschter Ton Vorbedingungen Software muss laufen, Android Client ist verbunden Nachbedingungen Ton ist abgespielt		Mittel	Hoch	NH

7 Aktivitätsdiagramm



8 Technische Machbarkeit

8.1 Technologien

Sämtliche Technologien, welche für das Projekt benötigt werden, sind in ähnlicher Form schon vorhanden und müssen nur mehr an das Projekt angepasst werden. Für alle möglichen Sensoren sind Codesamples vorhanden, welche die Einbindung in das Kuscheltier erleichtern. Allerdings gibt es auch bei diesem Projekt Risiken. So können zum Beispiel beim Multitasking der Sensoren Race-Conditions auftreten, welche das System zum Absturz bringen. Diese Fehler müssen durch synchronisieren der Aufrufe und intensives Testen beseitigt werden. Auch bei der Client → Server Verbindung zwischen Android App und Raspberry Pi können bei instabilen Verbindungen Probleme auftreten. Dies könnte Teile des Projekts zum Scheitern bringen.

8.2 Umsetzung

Die Umsetzung der Software erfolgt entweder in Python oder in Java. Am Raspberry Pi stehen standardmäßig beide Programmiersprachen zur Verfügung und können out-of-the-box eingesetzt werden. Die Android App wird in Java geschrieben, da Java standardmäßig auf Android eingesetzt wird. Sollte am Raspberry Pi Python zum Einsatz kommen, können Probleme bei der Kommunikation zwischen den beiden Programmiersprachen auftreten. Die benötigten Libraries für die GPIO Pins des Raspberry Pi werden von der Raspberry Pi Foundation für Python zur Verfügung gestellt und sind standardmäßig am Raspberry Pi installiert. Für Java gibt angepasste Libraries von Drittanbietern, wie zum Beispiel Pi4J. Wird die Software auch am Raspberry Pi mit Java umgesetzt, wird die Client → Server Kommunikation deutlich erleichtert.

Die Arbeitsumgebung ist die Konsole (SSH Verbindung) bzw. der Remote Debugger (Python)

8.3 Sensoren – Tests

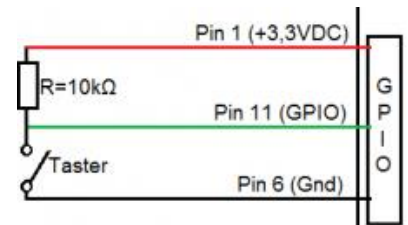
Die Ansteuerung der Ports erfolgte mit dem Command-Line-Tool „pigpio“, welches eine vereinfachte Ansteuerung ermöglicht.

Folgende Sensoren wurden getestet:

8.3.1 Taster

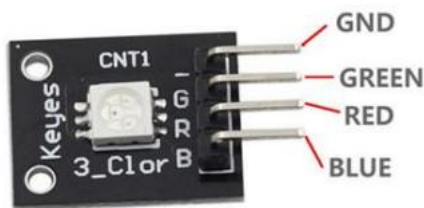
Beim Testen wurde ein Pullup – Widerstand verwendet, wodurch am GPIO- Port immer „high“-Signal (entspricht 1) anliegt. Wird er nun mit dem Ground – Port verbunden, ändert sich der Zustand auf „low“ (entspricht 0)

Der Taster wurde erfolgreich getestet.

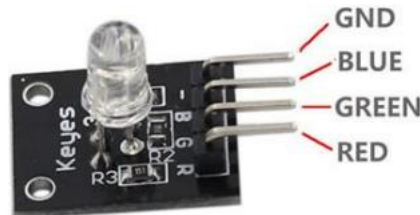


8.3.2 Kombi LED

Es wurde sowohl eine SMD (Surface Mount Device) – Kombi LED, als auch eine DIP (Dual In-line Package) – Kombi LED getestet.

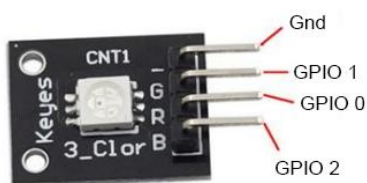


SMD Package



DIP Package

Beide LEDs wurden mit den gleichen Ports verbunden:



SMD Package



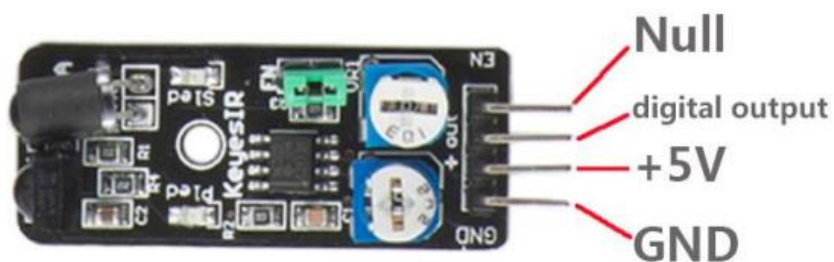
DIP Package

Über die Konsolen und dem „pigpio“ – Tool, wurden verschiedene RGB – Farben erfolgreich getestet.



8.3.3 Annäherungssensor

Der Annäherungssensor verwendet die Infrarot – Reflektion, um Hindernisse zu erkennen. Sollte kein Gegenstand davor sein, kann er kein Signal empfangen. Falls ein Gegenstand davor ist, wird das Infrarotlicht reflektiert und der Receiver kann ein Signal empfangen.

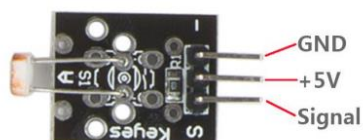


Der digitale Output wurde mit dem GPIO 0 – Port verbunden.

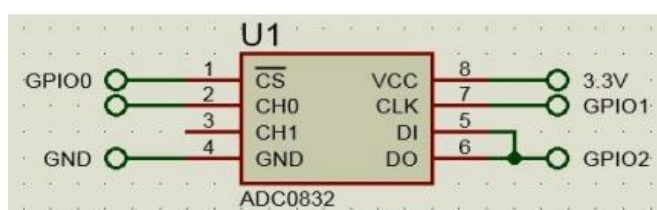
Die restlichen Anschlüsse wurden wie oben angeführt mit den Ports verbunden.

8.3.4 Helligkeitssensor

Der Helligkeitssensor misst die Helligkeit und gibt einen digitalen Wert aus.

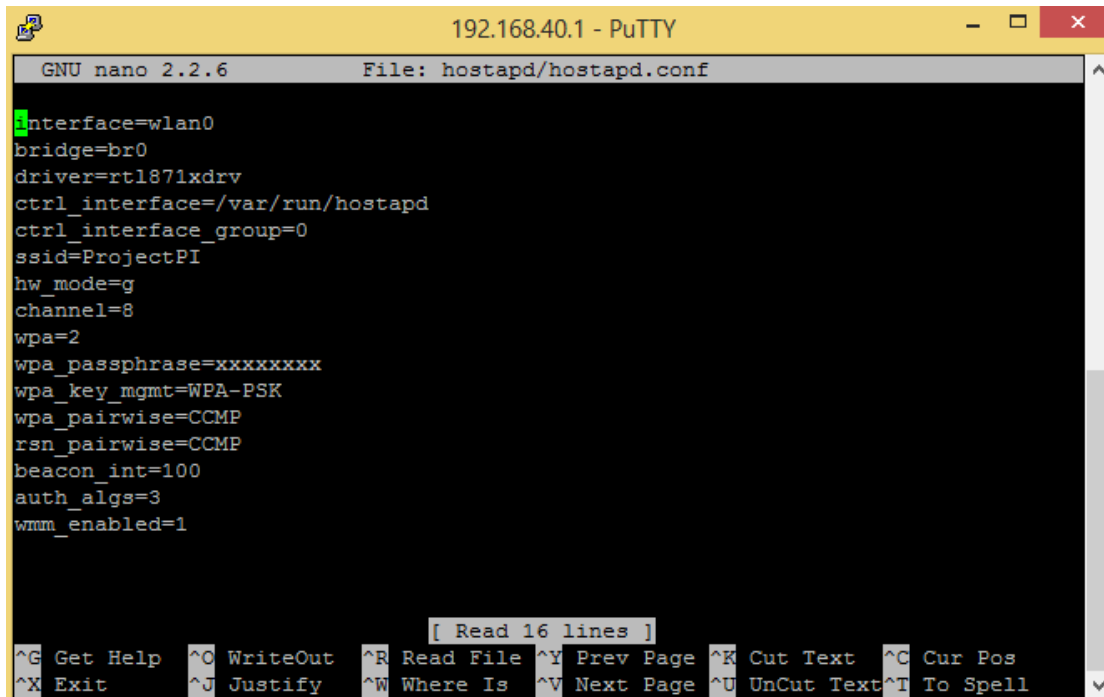


Die Ports wurden folgendermaßen verbunden:



8.3.5 WLAN – Kommunikation

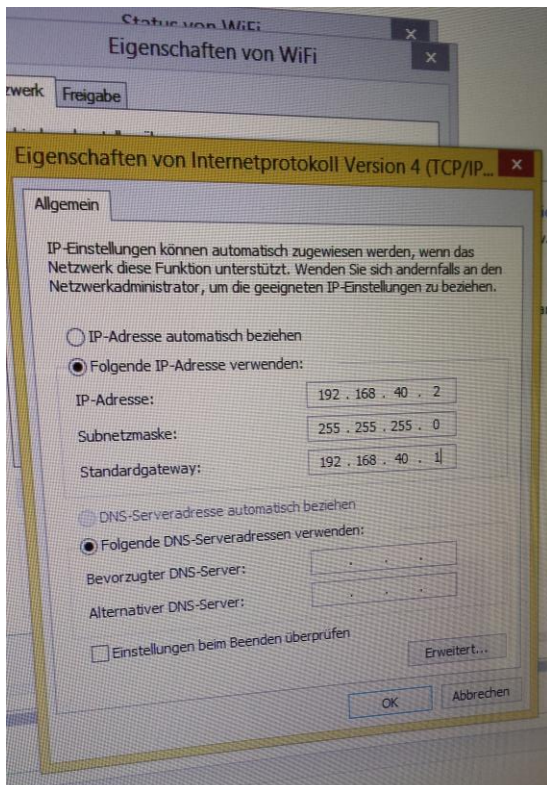
Die WLAN – Kommunikation wurde mit einem Realtek WiFi Dongle realisiert. So kann man sich mit dem Smartphone oder anderen Geräten (Notebook) mit dem Raspberry verbinden.



```
192.168.40.1 - PuTTY
GNU nano 2.2.6 File: hostapd/hostapd.conf
interface=wlan0
bridge=br0
driver=rtl871xdrv
ctrl_interface=/var/run/hostapd
ctrl_interface_group=0
ssid=ProjectPI
hw_mode=g
channel=8
wpa=2
wpa_passphrase=xxxxxxxx
wpa_key_mgmt=WPA-PSK
wpa_pairwise=CCMP
rsn_pairwise=CCMP
beacon_int=100
auth_algs=3
wmm_enabled=1

[ Read 16 lines ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Über die SSH – Verbindungen können Konfigurationen durchgeführt werden, auf Daten (SFTP) zugegriffen werden etc.



9 Wirtschaftliche Machbarkeit

9.1 Personalaufwand

Der Umfang der Software ist gering, somit müssen nicht alle Projektmitglieder an ein und derselbe Sache arbeiten. Die Arbeit kann deshalb gut eingeteilt werden, wodurch das Team schneller und effizienter an dem Projekt arbeiten kann. Durch das Verbinden über SSH können die beiden Programmieren gemeinsam an dem Raspberry arbeiten.

Da das Projekt mit 530 Stunden ein sehr umfangreiches Projekt ist, werden für eine effiziente Entwicklung vier Projektmitglieder am Werk sein. Diese Projektmitglieder verfügen über technisches Know-How, wodurch sehr professional auf die Sache herangegangen wird und das Projekt optimal umgesetzt wird.

9.2 Investitionsaufwand

Es gibt sehr geringe Investitionsaufwände für diverse Motoren, Sensoren und das Plüschtier. Für das restliche Hardwaresortiment, wie den Raspberry Pi, wurde uns von der Schule zum Entwickeln des Produkts übergeben. Die ganze Hardware gewährleistet erfolgreiches Arbeiten. Das benötigte Softwarepaket mit Python 3.3.5 sowie die Entwicklungsumgebung PyCharm wurden uns von unserem Auftraggeber zur Verfügung gestellt und fordern daher keine zusätzlichen Kosten.

Für den Router, welcher während der Entwicklungszeit zur Verfügung gestellt wird, musste die Gruppe 8€ zahlen. Weiteres mussten das Plüschtier („Pikachu“ – 30€), eine SD – Karte (8GB, Class 10 – 5€) und ein WLAN – Dongle (10€) eingekauft werden.

Da das Projekt im Bundesland Wien stattfindet, wo auch der Standort der Projektmitglieder ist, erfordert es keine Zusätzlichen Transport und Nächtigungskosten.

9.3 Nutzen

Das Resultat des Projektes ist für jeden Haushalt und jeden Kindergarten weltweit nützlich. Je nach Anfrage und Feedback von ausgewählten Personen wird entschieden, ob das Produkt auf den offenen Markt kommt oder nicht.

9.4 Risikoanalyse

9.4.1 *Personenausfall*

Es kann jederzeit ein Teammitglied in irgendeiner Form Ausfallen bzw. verhindert sein. In solchen Fällen müssen die Arbeitsaufgaben so verteilt werden, dass das Projekt nicht zu stark verzögert bzw. negativ beeinflusst wird.

Mögliche Gründe für einen Personenausfall sind:

- Krankheit
- Verhinderungen (familiäre Angelegenheiten, Schulungen)
- Austritt aus dem Projekt
- Auftraggeber zeigt keine Interesse mehr zur Umsetzung des Produktes

9.4.2 *Zeitliche Risiken*

Es kann jederzeit passieren, dass bei der Durchführbarkeit eine Fehleinschätzung eintritt. Dadurch kommt es zu einer Zeitverschiebung der Meilensteine und logischerweise zu einer höheren Auslastung der Mitarbeiter. Weiteres kann es durch externe Arbeiten der Projektmitglieder zu Zeitverzögerungen kommen.

9.4.3 *Technische Risiken*

- **Datenverlust**

Es kann jederzeit zu unerwarteten Komplikationen wie Datenverlust kommen. Auslöser können Userfehler oder z.B. Viren sein. Eine gute Gegenmaßnahme bei Datenverlust sind Backuplösungen, welche automatisiert erstellt werden.

- **Serverausfälle**

Auch bei den Servern (Web – Blog) besteht die Möglichkeit, dass sie aufgrund verschiedener Ursachen ausfallen. Sei es jetzt bei einem Stromausfall oder bei einer DDoS – Attacke. Wichtig ist, dass bei einem Stromausfall die Daten wiederhergestellt werden (siehe oben) und ein DDoS – Schutz wie z.B. den „Kona Site Defender“ verwendet wird.

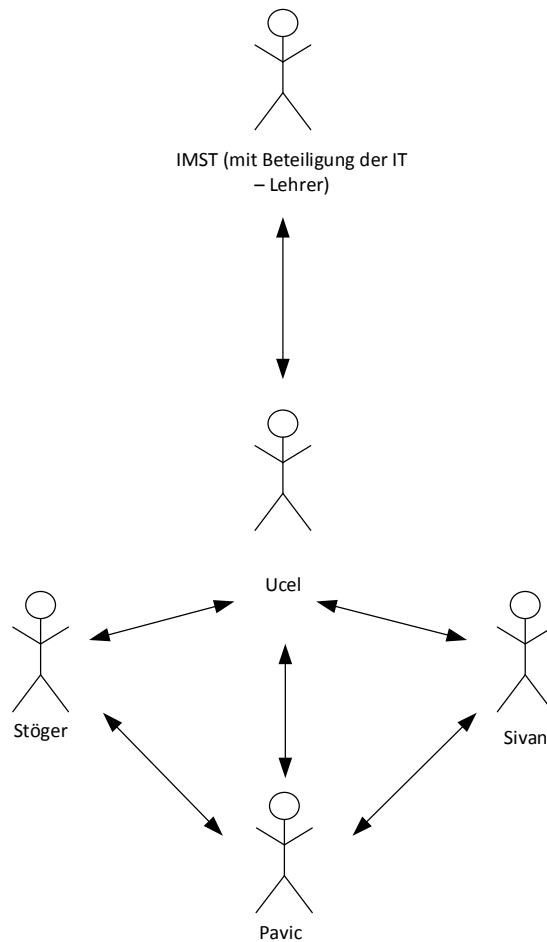
10 Nutzwertanalyse

Ziel	Kriterien	Gewichtung in %	Eigene		Fremde (Libraries)	
			Note	Teilnutzwert	Note	Teilnutzwert
App	Betriebskosten (Strom etc.)	10%	1	10	1	10
	Wartung	10%	2	20	2	20
	Anschaffung	10%	4	40	1	10
	Ressourcen (Erweitbarkeit)	15%	1	10	3	30
	Support	5%	2	20	2	20
Summe		50%	2	100	1,8	90
Software (Raspberry)	Lizenzkosten	10%	1	10	4	40
	Arbeitsaufwand	15%	5	50	2	20
	Anpassbarkeit	10%	3	30	3	30
	Support	5%	2	20	2	20
	Kompatibilität	10%	4	40	3	30
Summe		50%	3	150	2,8	140

Gesamtbewertung	250	230
Endreihung	2	1

Wie man hier in der Nutzwertanalyse erkennen kann, wäre eine Kombination von fertigen Libraries und eigener Software am besten. Dadurch erspart man sich zusätzliche Programmierungen und hat eine verlässlichere Kompatibilität. Bei der eigenen Software hat man den Vorteil, dass man sich die Lizenzkosten erspart und seine Software „besser kennt“. Dies ist vor allem bei Anpassungen oder Erweiterungen der Software sehr hilfreich, jedoch ist der Arbeitsaufwand dementsprechend höher.

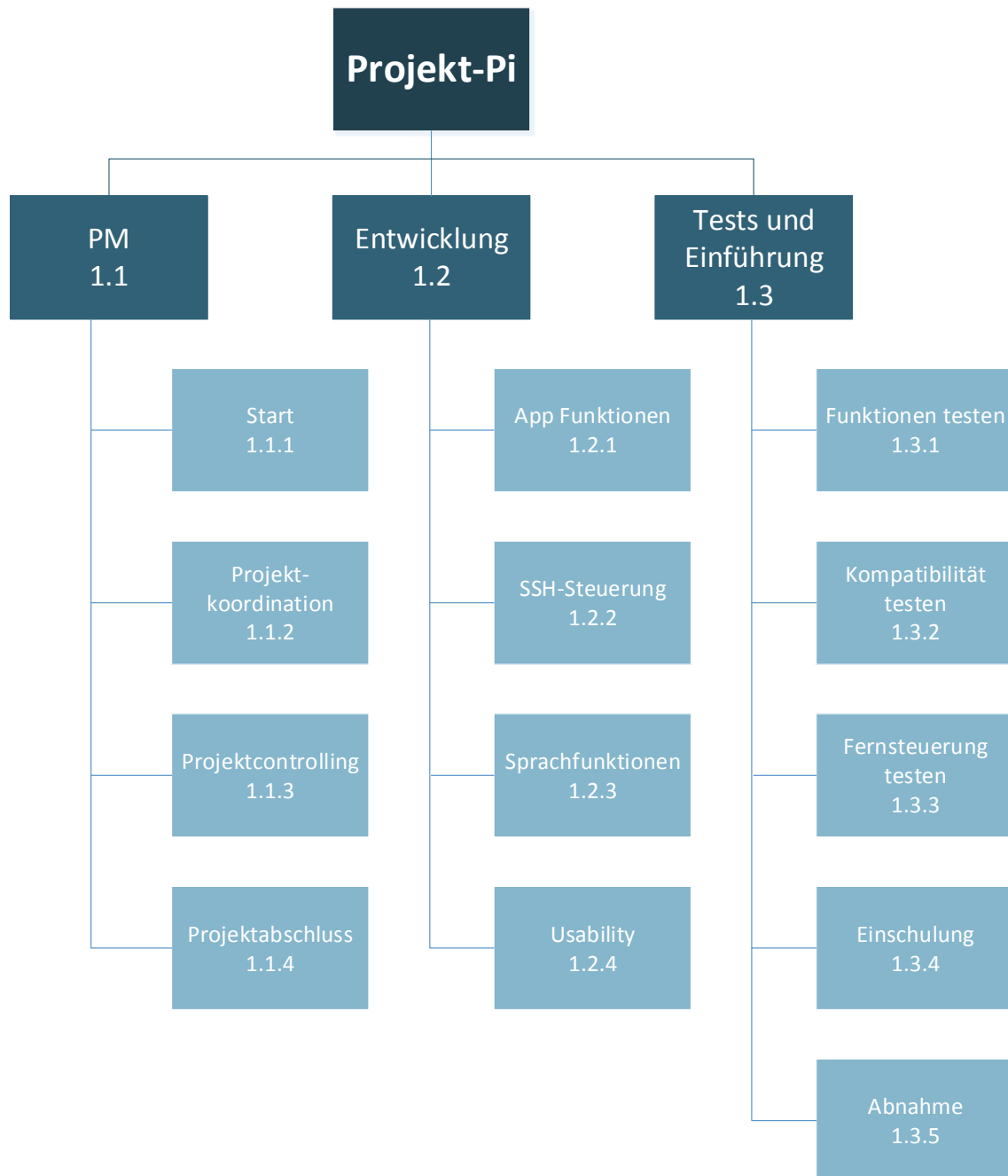
11 Projektorganisation



Das Projektteam besteht aus:

- Dem Projektleiter – Ucel
- Den Projektmitgliedern: Sivan, Pavic, Stöger

12 Projektplanung



Meilenstein	Datum
Komponenten erhalten	09.02.2015
Schaltplan fertigstellen	16.02.2015
Programmierung abgeschlossen	09.03.2015
Testen abgeschlossen	20.03.2015
Vorbereitung des Plüschtiers abgeschlossen	31.03.2015
Einbauen und Verkabeln der Komponenten abgeschlossen	15.04.2015
Projektabschlussbericht übergeben	11.05.2015

13 Management Summary

Die IT – Abteilung des TGMs in Wien, möchte in Kooperation mit dem IMST, ein sensibles Kuscheltier entwerfen.

Zurzeit gibt es auf dem Markt einige Alternativen, jedoch sind das nur „Teillösungen“. Das Projekt „Project PI“ bietet ein funktionsfähiges und erweiterbares System. Das System kann sowohl auf einem eigenen Server, also auch auf einem gemieteten (siehe Nutzwertanalyse) betrieben werden. Die Software des Projektes kann sowohl eigene Software, als auch mit fertigen Libraries realisiert werden. (siehe Nutzwertanalyse)

Auch die Umsetzung auf eine mobile Seite ist problemlos machbar, wobei das System für Android-Devices optimiert wird in Form einer App. (größter Marktanteil)

Das komplette Projekt wird ca. 3-4 Monate benötigen und ist sowohl technisch als auch wirtschaftlich machbar.