Projektplan

Inhaltsverzeichnis

1 Einführung

1.1 Zweck des Dokuments

Ziel des Dokuments ist es, dem Leser einen Überblick über die Studienarbeit zu geben und so einen schnellen Einstieg in den Projektablauf zu gewähren. Dabei wird der Projektablauf definiert, die möglichen Risiken analysiert und ein Überblick über die Arbeitsweise und Infrastruktur gegeben.

1.2 Referenzen

Dieses Dokument dient als Ergänzung zum technischen Bericht der Studienarbeit. Alle projektspezifischen Informationen, die im Bericht keinen Platz finden, sind hier aufgeführt.

2 Projektübersicht

Motivation, Zweck und Ziel, Lieferumfang, Annahmen und Einschränkungen werden im technischen Bericht bereits detailliert ausgeführt.

Um allen Beteiligten einen möglichst einfachen Zugriff zum aktuellen Projektstand zu ermöglichen, wurde eine Webseite erstellt. Dort sind die wichtigsten Dokumente und weiterführende Links aufgelistet.

Webseite: http://flux-coordinator.com

3 Projektorganisation

Diese Arbeit wird als Bachelorarbeit an der HSR Hochschule für Technik Rapperswil im Frühjahrssemester 2018 durchgeführt.

3.1 Organisationsstruktur

Diese Bachelorarbeit findet im Auftrag der Firma *HSi Elektronik AG* statt. Der Betreuer für das Projekt ist ein Dozent der Hochschule.

3.1.1 Studierende

- Esteban Luchsinger, esteban.luchsinger@hsr.ch
- Patrick Scherler, patrick.scherler@hsr.ch

3.1.2 Betreuer HSR

- Prof. Dr. Farhad Mehta, *Institut für Software*, farhad.mehta@hsr.ch

3.1.3 Ansprechpartner Auftraggeber

- Tobias Hofer, *HSi Elektronik AG*, tobias.hofer@hsi-ag.ch

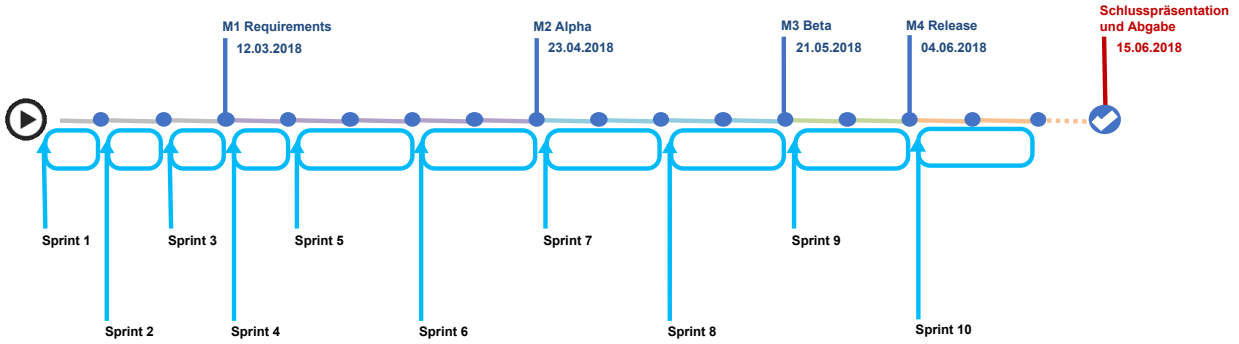
4 Management Abläufe

4.1 Zeitliche Planung

Die zeitliche Planung ist grösstenteils durch die Organisation als Bachelorarbeit gegeben. Insgesamt sollen ca. 360 Arbeitsstunden pro Student investiert werden. Folgende Termine sind definiert:

Termin	Beschreibung	Betrifft
19.02.18	Beginn der Bachelorarbeit	Studierende
23.02.18	Kickoff Meeting Auftraggeber	Stud. & Auft.
26.02.18	Kickoff Meeting Betreuer	Stud. & Betreuer
08.06.18	Abgabe von Abstract und Poster an Betreuer	Studierende
13.06.18	Der Betreuer gibt das Dokument mit dem korrekten und vollständigen Abstract zur Weiterverarbeitung an das Studiengangsekretariat frei.	Betreuer
15.06.18	Präsentation und Ausstellung der Bachelorarbeiten, 16 bis 20 Uhr	Alle
15.06.18	Abgabe des Berichts an den Betreuer bis 12:00 Uhr	Studierende
Ab 15.06.18	Mündliche BA-Prüfung	Studierende

4.2 Vorgehen



4.3 Meilensteine

M1 Requirements

Die Anforderungen sind definiert und mit allen Beteiligten abgesprochen.

Work Products: Use-Case Diagramme, Mockups, Prototypen

M2 Alpha

Die Architektur des Systems steht fest und ist implementiert. Es gibt einen «Durchstich» durch alle Komponenten des Systems. TBD

Work Products: Executable Architecture

M3 Beta

Dieser Meilenstein markiert den Übergang in die Stabilisationsphase des Projekts. Alle Features sind zu diesem Zeitpunkt implementiert. Ab hier liegt der Fokus auf Bugfixing und Analyse (Performance, UX, etc.). TBD

Feature Freeze: Ab diesem Meilenstein werden keine grösseren Features mehr eingeführt.

Work Products: Executables für Server und Clients

M4 Release

Die Applikation kann an den Auftraggeber ausgeliefert werden.

Code Freeze: Ab diesem Meilenstein gibt es einen Code-Freeze. Es sollten keine offensichtlichen Fehler mehr vorhanden sein. TBD

Work Products: Executables für Server und Clients

MP Abgabe und Präsentation

Der letzte Meilenstein ist die Präsentation der Arbeit. TBD

Work Products: Bericht, Poster, Präsentation und Demo

4.4 Besprechungen

Besprechungen werden mit dem Betreuer regelmässig geplant. Dem Auftraggeber wird regelmässig per E-Mail, Telefonkonferenz oder persönlich Bericht erstattet. Beim Auftauchen von Problemen, sind diese mit dem Auftraggeber zu lösen und erst falls eine Eskalation notwendig ist, soll der Betreuer hinzugenommen werden.

4.4.1 Protokollführung

Alle Besprechungen werden mit einer Traktandenliste vorbereitet und den Teilnehmern wenn möglich 24 Stunden vorher zugestellt.

Zu allen Besprechungen werden Protokolle geführt und auf der Projektwebsite passwortgeschützt veröffentlicht.

5 Risikomanagement

Die Risikobeurteilung führt alle erfassten projektspezifischen Risiken als Produkt aus der geschätzten Eintrittswahrscheinlichkeit und dem erwarteten Schadensausmass in Stunden auf.

5.1 Risiken

#	Titel	Beschreibung	Massnahme	Ausmass [h]	Wahrschein-lichkeit [%]	Risiko [h]
1	Geringe Leistung des Raspberry Pi	Die Hardware des Raspberry Pi Zero (RAM, CPU, Schnittstellen) reicht für das Projekt nicht aus.	- Mit Architektur-Prototyp testen - Erneute Evaluation der Hardware	16	40	6.4
2	Lux-Sensor Schnittstelle nicht vorhanden oder dokumentiert	Die Lux-Sensor Werte lassen sich nicht über eine dokumentierte Schnittstelle auslesen.	- Funktionstests zu Projektbeginn - Risiko mit Kunde besprechen	40	60	24
3	Pozyx entspricht nicht den Anforderungen	Stabilität, Genauigkeit oder Performance von Pozyx reichen nicht aus.	- Funktionstests zu Projektbeginn - Risiko mit Kunde besprechen	40	40	16
4	Ausfall / Verlust der Projekthardware	Die für die Umsetzung benötigte Hardware (Raspberry Pi, Pozyx, Lux-Sensor) steht nicht mehr zur Verfügung.	- Hardware-Lieferzeiten beachten - Hardware einschliessen - ESD-Massnahmen beachten	16	30	4.8
5	Probleme mit Projektinfrastruktur	Datenverlust oder fehlende Verfügbarkeit der Management Tools	- Self-hosting - automatisches Backup	16	20	3.2
6	Verzögerungen bei der Entwicklung der Benutzerschnittstelle	Verzögerungen durch fehlende Praxiserfahrung	- Zeit für eine detaillierte Einarbeitung in die Technologie einplanen	16	30	4.8

5.2 Umgang mit Risiken

Die grössten Risiken sollen bereits im ersten Drittel des Projekts minimiert werden. Dazu werden Funktionstests durchgeführt und Prototypen erstellt, um bei Problemen möglichst schnell reagieren zu können.

Mit dem Auftraggeber wurde ausserdem besprochen, dass der Fokus dieser Arbeit in der Software-Entwicklung und Visualisierung der Sensordaten liegt. Die Ansteuerung der Sensor-Hardware oder andere nicht Software-spezifische Aufgaben können notfalls dem Auftraggeber übergeben werden.

6 Arbeitspakete

Die Arbeitspakete, Meilensteine und Sprints werden auf Jira verwaltet.

http://jira.flux-coordinator.com

Der normale Workflow der Arbeitspakete wurde im Zustandsdiagramm unten abgebildet.

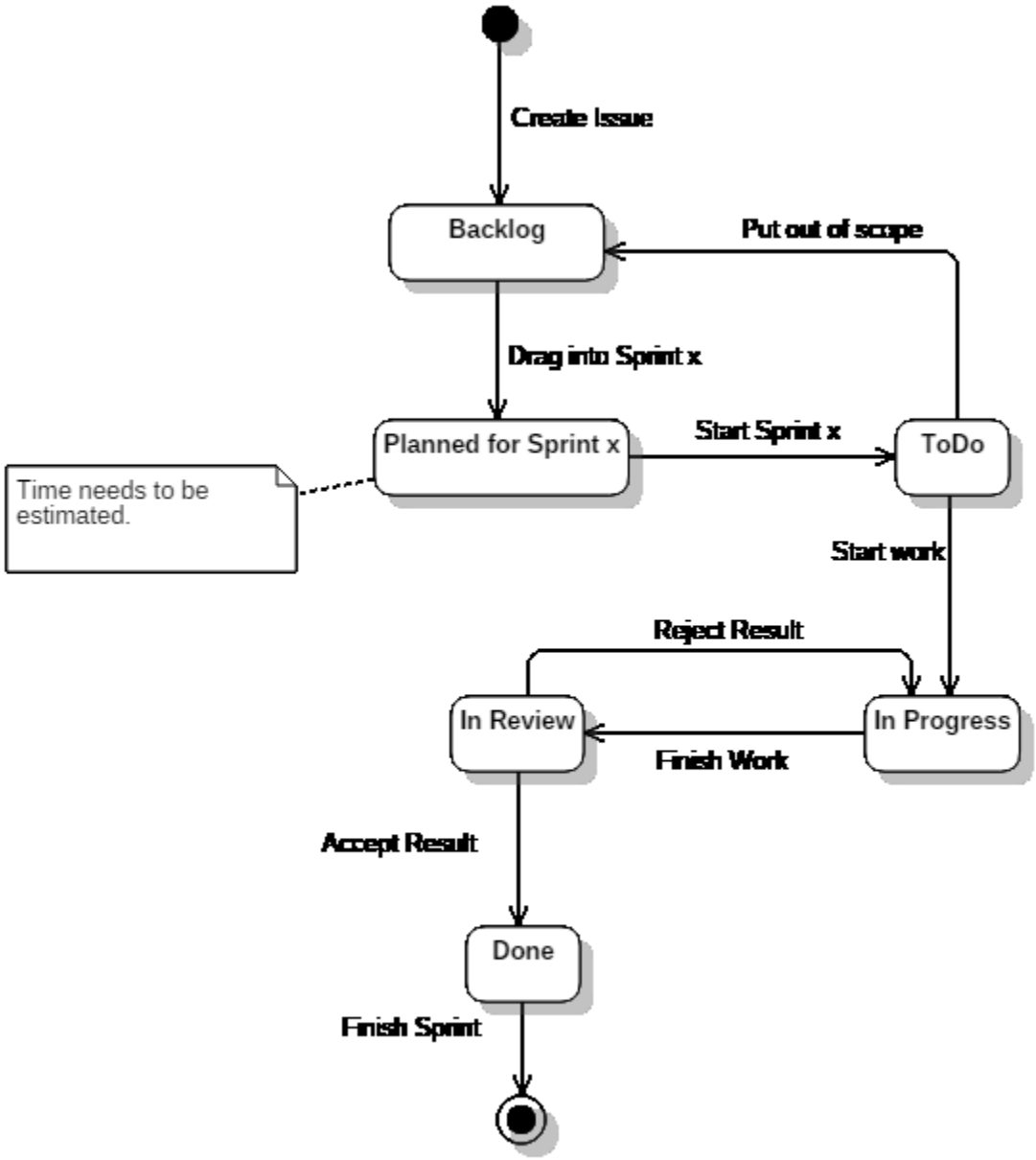


Abbildung 1 Zustandsdiagramm für die Arbeitspakete in Jira

7 Infrastruktur

In diesem Kapitel wird die für diese Arbeit verwendete Infrastruktur aufgelistet und erklärt.

7.1 Projektmanagement Server

Für das Hosten der Projektmanagement Software («Jira») wird ein Server mit Ubuntu verwendet. Dieser wird von der HSR bereitgestellt und In-House gehostet.

7.2 Version Control System

Der Code der verschiedenen Projekte wird auf einem Git VCS gehostet. Der gewählte Anbieter übernimmt auch das Betreiben der benötigten Infrastruktur.

7.3 Hosting der Server-API

Das Projekt wird eine API für die Kommunikation zwischen den verschiedenen Clients zur Verfügung stellen. Diese soll Plattformunabhängig sein und auch auf der «Cloud» gehostet werden können. Zum Hosting der Anwendung wird daher auch ein Server verwendet. Der Server wird vom gewählten Cloud Anbieter gehostet.

7.4 Continuous Integration & Continuous Delivery

Für das Testing der Anwendung muss ein CI/CD Dienst aufgesetzt werden. Das Hosting der Testinfrastruktur wird vom CI/CD Anbieter übernommen.

7.5 Lichtmessgeräte

Die Lichtmessungen erfordern die Verwendung von spezialisierten Messgeräten. Diese heissen im Fachjargon «Luxmeter». Für die Anwendung wird ein vom Auftraggeber gestellter TCS3430EVM Sensor des Herstellers AMS mit einer USB/I2C Schnittstelle verwendet. Dieser Sensor kann abgesehen von den Luxwerten auch die Farbtemperatur und Farbe messen.

Im Verlaufe der Arbeit kann es nötig sein, einen zweiten Sensor oder ein dediziertes Luxmeter anzuschaffen, um den TCS3430EVM zu kalibrieren. Dieser kann beim Auftraggeber ausgeliehen oder verwendet werden.

7.6 Positionierungssystem

Für die genaue Positionsbestimmung wurde der «Developer's Kit» der Firma Pozyx angeschafft. Dieses System ermöglicht die genaue Positionsbestimmung im Zentimeterbereich.

Zum System gehören folgende Komponenten:

- Bitte alle Komponenten auflisten, die in der Kiste dabei waren.

7.7 Weitere Infrastruktur

Zusätzlich zu der oben erwähnten Infrastruktur wurde ein Raspberry Pi Zero und ein Arduino gekauft. Diese sollen die Kommunikation mit den Sensoren bewerkstelligen.

Ebenfalls kann bei der Arbeit ein Hilfssystem zur Bewegung der Sensoren verwendet werden. Das Hilfssystem muss während der Arbeit evaluiert und beschafft werden.

8 Qualitätsmassnahmen

Durch die in diesem Kapitel behandelten Massnahmen wird die Qualität der abgelieferten Software sichergestellt.

8.1 Entwicklungs-Workflow

Es wird nach dem *Gitflow Workflow* entwickelt. Anstatt einen einzigen master-Branch zu verwenden, werden feature-Branches in einen development-Branch zusammengelegt. Die dadurch entstehende Stabilisierung des master-Branches wirkt sich positiv auf eine mögliche Continuous Delivery Pipeline aus. So ändert sich eine angebotene API nicht bei jedem Commit in den development-Branch und die Releases können besser geplant werden.

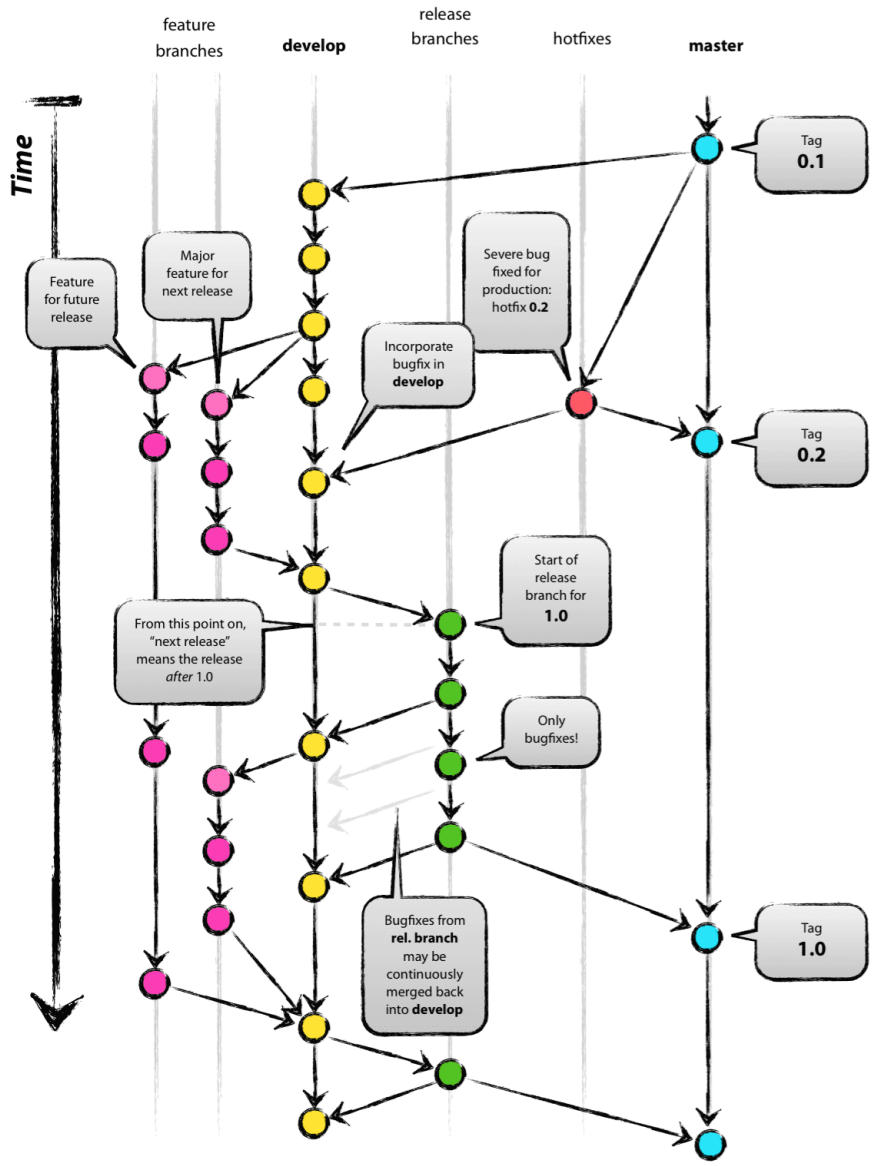


Abbildung 2 Vincent Driessen, Creative Commons BY-SA

8.2 Code Style

Da in dieser Arbeit mehrere Programmiersprachen und Werkzeuge verwendet werden, kann eine einheitliche Coding Guideline erfolgen. Das wichtigste Element des Coding Stils ist jedoch, dass der Programmierstil innerhalb einer Technologie einheitlich ist. Bei jeder neu verwendeten Programmiersprache muss beim Aufsetzen des Projektes ein Coding Styleguide ausgewählt werden und durchgezogen werden. Die verwendeten Coding Styleguides müssen pro Programmiersprache sauber vermerkt werden.

8.3 Testing

Einer der Grundsätze dieser Arbeit ist, dass alle Komponenten automatisiert getestet werden müssen. Für Webserver gibt es eine komplette CI/CD Pipeline, welche an das VCS angeschlossen wird. Beim Pushen von Commits auf den VCS Dienst und vor dem Zusammenlegen von Branches, wird der CI Server die Tests automatisch durchführen und das Ergebnis anzeigen. Ein Zusammenlegen von Branches ohne die Tests zu bestehen ist nicht erwünscht und wird durch Massnahmen des VCS verhindert.

Eine Ausnahme des automatischen Testing bilden die zum Teil relativ hardwarenahen Komponenten, die mit den Sensoren kommunizieren. Diese können nicht komplett automatisiert getestet werden. Vielleicht ist es jedoch möglich, wenigstens einen Teil der hardwarenahen Software mit Hilfe von Sensor-Mocking zu testen.

8.4 Definition of Done

Die Definition of Done (DoD) ist die Messlatte, an der die Fertigstellung eines Produktes gemessen wird. Wenn die Kriterien der DoD nicht eingehalten sind, ist das Produkt noch nicht fertig. Wichtig dabei ist, dass die Kriterien nicht schwarz-weiss, sondern im Kontext verstanden werden müssen. Auch die verschiedenen Interpretationsmöglichkeiten lassen oft keine klare Linie zwischen Einhaltung und Nichteinhaltung des DoD ziehen.

Folgende sind die einzuhaltenden Kriterien:

Akzeptanzkriterien	Die Akzeptanzkriterien, welche in den User Stories definiert wurden, sind erfüllt.
Code Styling	Die Richtlinien für den Code sind erfüllt. (Siehe 8.2)
CI Tests	Die Tests, die auf dem Continuous Integration Dienst ausgeführt werden sind erfüllt. (Siehe 8.3)
Review	Bei einigen kritischen Funktionen kann ein Code Review erwünscht / benötigt sein. Falls die Funktion im Voraus für Code Review markiert wurde, muss dieser durchgeführt worden sein.

Falls die Kriterien für ein Feature nur mit übermässigem Aufwand vollständig erfüllt werden können, können Ausnahmen mit dem Auftraggeber und Betreuer diskutiert werden. Dabei sollten aber auch die Auswirkungen von technical debts diskutiert werden und eine klare Schuldengrenze gezogen werden.