Requirements / Design and Test Dokumentation

(RDT)

Version 1.0

ESEP – Praktikum – SoSe 2024

Team – ESEP-1.2

Mjawaz, Yousef, 2636830, yousef.mjawaz@haw-hamburg.de

Althiab, Hadi, 2637134, hadi.althiab@haw-hamburg.de

Shaya, Hussam, 2552404, hussam.shaya@haw-hamburg.de

Shahrour, Samer, 2587833, samer.shahrour@haw-hamburg.de

Inhaltsverzeichnis:

[1 Teamorganisation 4](#_Toc170397188)

[1.1 Verantwortlichkeiten 4](#_Toc170397189)

[1.2 Absprachen 4](#_Toc170397190)

[1.3 Repository-Konzept 4](#_Toc170397191)

[2 Projektmanagement 5](#_Toc170397192)

[2.1 Qualitätssicherung 5](#_Toc170397193)

[3 Problemanalyse 5](#_Toc170397194)

[3.1 Analyse des Kundenwunsches 5](#_Toc170397195)

[3.1.1 Stakeholder 5](#_Toc170397196)

[3.1.2 Anforderungen 5](#_Toc170397197)

[3.1.3 Use Cases 7](#_Toc170397198)

[3.2 Anlage: Analyse der technischen Gegebenheiten 9](#_Toc170397199)

[3.2.1 Technischer Aufbau und Hardwarekomponenten 9](#_Toc170397200)

[3.3 GPIO 10](#_Toc170397201)

[3.3.1 Werkstücke 11](#_Toc170397202)

[3.4 Softwareebene 11](#_Toc170397203)

[3.4.1 Schnittstellen: Nachrichten und Signale 11](#_Toc170397204)

[4 Software-Design 14](#_Toc170397205)

[4.1 Software Architektur 14](#_Toc170397206)

[5 Implementierung: Besonderheiten 15](#_Toc170397207)

[6 Qualitätssicherung 16](#_Toc170397208)

[6.1 Teststrategie 16](#_Toc170397209)

[6.2 Testszenarien/Abnahmetest 16](#_Toc170397210)

[7 Technische Schulden 19](#_Toc170397211)

[8 Remote Control: 20](#_Toc170397212)

[9 Betriebsanleitung 21](#_Toc170397213)

[9.1 Kalibrierungsmodus: 21](#_Toc170397214)

[9.2 Betriebsmodus: 22](#_Toc170397215)

[10 Modellierung/FSMs Konzept 22](#_Toc170397216)

[11 Lessons Learned 23](#_Toc170397217)

[12 Anhang 23](#_Toc170397218)

[12.1 Abkürzungen 23](#_Toc170397219)

[12.2 Fragen, die Während des Projekts aufkamen 24](#_Toc170397220)

Abbildungsverzeichnis:

[Abbildung 1: Repository-Konzept, Quelle: Prof. Lehmann 4](#_Toc170397543)

[Abbildung 2: Skizze der Anlagen, zeigt die vorhandenen Sensoren und deren Platzierung/kürzel 10](#_Toc170397544)

[Abbildung 3: GPIO-PINS 10](#_Toc170397545)

[Abbildung 4: Richtige Werkstücke 11](#_Toc170397546)

[Abbildung 5: Sequenzdiagramm. 13](#_Toc170397547)

[Abbildung 6: Software-Architektur 14](#_Toc170397548)

# Teamorganisation

## Verantwortlichkeiten

|  |  |
| --- | --- |
| Verantwortlichkeit | Person/en |
| *MQTT (Broker & Oberfläche)* | *Hussam* |
| *Interrupts (Konzept, Code & Test)*  *Encoder (Konzept, Code & Test)*  *Dispatcher (Konzept, Code & Test)*  *Message Passing (Konzept, Code & Test)*  *QNet (Konzept, Code & Test)*  *FSMs (Modellierung & Implementierung)* | *Hadi,*  *Samer,*  *Yousef* |

## **Absprachen**

* Sonntags Online-Meeting via Discord ab 11 Uhr
* Terminabsagen müssen frühzeitig in der WhatsApp-Gruppe angekündigt werden.

## Repository-Konzept

Ein Bild, das Text, Screenshot, Reihe, Schrift enthält.

Automatisch generierte Beschreibung

Abbildung 1: Repository-Konzept, Quelle: Prof. Lehmann

Wie in der Vorlesung gezeigt: Es wird ein Main Branch geben, auf dem eine Lauffähige Version zu finden ist. Ein anderer Branch ist der Development Branch, auf dem Features und deren Integration getestet werden.

Diese beiden Branches sind online im Git-Repository verfügbar. Es werden aber auch andere lokale Branches vorhanden sein, auf denen Features aktiv implementiert und überarbeitet werden. Wenn ein Feature fertig implementiert wurde, dann wird dieser auf den Development Branch gemerged.

# Projektmanagement

## Qualitätssicherung

* Beim Bearbeiten dieses Dokuments benutzt jedes Teammitglied eine andere Farbe für seinen selbstgeschriebenen Text. Erst wenn die Mehrheit der Teammitglieder diesen Text gelesen hat und mit dem Inhalt einverstanden ist, wird dieser Text in die Farbe schwarz umgewandelt.
* Es wurde auf eine Sprache beim Codieren geeignet: Englisch.
* Bei Zweifel (nach der Modellierung) haben wir neu angefangen und versucht eine Variante zu finden, mit der die Mehrheit zufrieden war.

# Problemanalyse

## Analyse des Kundenwunsches

### Stakeholder

|  |  |
| --- | --- |
| Stakeholder | Interessen |
| HAW | Die Anlagen sollen nicht kaputt gehen |
| Prof. Lehmann/ Enrico | Die Aufgabe muss erfüllt sein. Damit das Team die PVL bekommt. |

### Anforderungen

|  |  |
| --- | --- |
| Nr. / ID | Beschreibung |
| **R1: Sortieren** | R1.1: Das System muss WPs voneinander unterscheiden können. |
| R1.2: Reihenfolge: mit Bohrung mit Metall → mit Bohrung mit Metall → flaches Werkstück (12) |
| R1.3: Aussortierte WPs (defekte oder nicht in der richtigen Reihenfolge) fallen in die Rutsche. (16,18) |
| R1.4: Wenn FB1 nicht aussortieren kann, dann soll auf FB2 versucht werden zu sortieren. (17) |
| R1.5: wird auf dem FB2 ein anderes WP oder eine andere Höhe erkannt als auf FB1, so soll das WP aussortiert werden. (19) |
| R1.6: Daten, wie WP-ID, Typ und mittlere Höhe in mm vom WP sollen erfasst werden. (28-31) |
| R1.7: Der Typ von einem WP ist im Regalfall am Ende von FB1 bekannt. (33) |
| R1.8: Das System vergibt WP-ID am Anfang von FB1.(32) |
| **R2: Förderband** | R2.1: WPs können gelegt werden, wenn der Anfang vom FB1 frei ist. (21) |
| R2.2: Transport auf beiden Laufbändern erfolgt während Höhenmessung nur langsam. (24) |
| R2.3: Es darf kein WP vom FB fallen. (25) |
| R2.4: Laufbänder stoppen, wenn kein WP darauf ist. (34) |
| R2.5: Laufbänder stoppen, wenn ein Fehler auftritt (Siehe R5). |
| R2.6: FB2 stoppt, wenn ein WP am Ende von FB2 ankommt (WP bleibt auf FB2). (28\_0) |
| R2.7: Möglichst höher Durchsatz an WPs soll erreicht werden. (35) |
| **R3: Anzahl WPs** | R3.1: Mehrere WPs dürfen sich auf FB1 befinden. (22) |
| R3.2 Es darf sich maximal ein WP auf FB2 befinden. (23) |
| **R4: Ampel, LEDs, Signalisierung** | Es muss an der jeweiligen Anlage signalisiert werden, wenn:   * R4.1: Betreibzustand läuft **(Ampel Grün dauerhaft)**. (73) * R4.2: Service-Mode gestartet wurde **(Ampel Grün blinkend)**. (74) * R4.3 Warnungen auftreten **(Ampel Gelb)**. (Siehe R5) (75) * R4.4: Fehler auftreten **(Ampel Rot).** (Siehe R5, R8) |
| **R5: Fehler/Warnung Erfassung** | Es gibt einen Fehler, wenn:   * R5.1: Beide Rutschen voll sind und ein notwendiges Aussortieren ist nicht mehr möglich ist. (56) * R5.2: Ein WP verschwindet. (54) * R5.3: Ein WP wird außerhalb des Anfangsbereiches hinzugefügt. (55) * R5.4: der Mindestabstand zwischen den WPs nicht eingehalten wurde. * R5.5: mehrere WPs sich auf dem FB2 befinden.   Es gibt eine Warnung, wenn:   * R5.6: eine Rutsche voll ist oder wenn beide Rutschen voll sind. |
| **R6: Bedientaster** | R6.1: **Start**: in Betriebszustand wechseln (wenn kurz gedrückt).(62)In Service-Mode wechseln (wenn lange gedrückt) -> Selbsttest/Kalibrierung. (61) |
| R6.2: **Stop**: in Ruhezustand wechseln (62), wenn kein Fehler oder Warnung (63). |
| R6.3: **Reset**: Fehlerquittierung. (64) |
| R6.4: **E-Stop**: Die beiden Anlagen stehen still (65), bis der E-Stopp-Schalter herausgezogen und dann der Reset-Taster gedrückt wurden (67). |
| **R7: Remote Control** | R7.1: Die Anlage soll mit einem Remote-System gesteuert und überwacht werden (105). |
| R7.2: Die Kommunikation soll auf MQTT basieren (107). |
| R7.3: Die Remote-Control-Anwendung soll eine graphische Oberfläche haben (108). Entweder eine native Anwendung oder eine Browser Anwendung (109). |
| R7.2: Die Button-Panel beider Anlagen ist in der Oberfläche sichtbar und benutzbar (107). |
| R7.3: Relevante Informationen (z.B. Konsolen ausgaben, volle Rutsche) sollen auch in der Anwendung ausgegeben werden (111-112). |
| **R8: Fehlerzustände** | R8.1: Kein Fehler -> Rote Lampe aus (89). |
| R8.2: Wenn ein Fehler neu auftritt, geht die Anlage in den Zustand „Anstehend unquittiert“ -> rote Lampe blinkt schnell (1 HZ) (78). Bis Reset-Taste gedrückt wurde (79) |
| R8.3: Wenn die Anlage im Zustand „Anstehend unquittiert“ ist und der Reset-Taster gedrückt wurde, geht die Anlage in den Zustand „Anstehend quittiert“ -> rote Lampe leuchtet dauerhaft (91). |
| R8.4: Wenn ein Fehler automatisch vom System behoben wird, dann geht die Anlage in den Zustand „Gegangen unquittiert“ -> rote Lampe blinkt langsam (0.5 Hz) (92), bis Reset-Taste gedrückt wurde (85) |

|  |  |  |
| --- | --- | --- |
| Unterschiede zwischen den Anlagen | Anlage 1 | Anlage 2 |
| Mehrere WPs gleichzeitig | ✔️ | ❌ |
| Vergeben von WP-ID | ✔️ | ❌ |
| Überprüfung der Übereinstimmung des geschickten WPs | ❌ | ✔️ |
| WP stoppt bei LBE | ✔️, bis kein WP sich auf Anlage 2 befindet | ✔️, bis der Roboter das WP entnimmt |

### Use Cases

|  |  |
| --- | --- |
| **Titel** | Prüfprozess eines Werkstückes |
| **ID** | U1 |
| **Akteure** | Kunde, WP, Anlage |
| **Ziel** | Es wird überprüft, ob ein WP bei LBE2 vom Roboter entnommen werden kann. |
| **Auslöser** | Der Kunde legt ein WP bei LBB1 |
| **Vorbedingung** | beide Anlagen laufen fehlerfrei. Anlagen sind im Betriebszustand, kein WP befindet sich auf FB1/2. |
| **Nachbedingung** | Das Werkstück ist entweder aussortiert oder vom Roboter entnommen worden. |
| **Erfolgsszenario** | 1. Kunde legt ein WP bei LBB1. 2. Das WP bekommt eine ID vom System zugewiesen. 3. WP kommt beim HS an und FB1 wird langsamer 4. HS fängt an, Daten über WP zu erfassen. 5. WP kommt bei LBHS1 an und unterbricht diese. 6. Das WP kommt beim MS an. 7. MS fängt an, Daten über WP zu erfassen 8. WP kommt bei LBS1 und unterbricht diese. 9. Das System prüft die vorher gemessenen Daten (in Schritt 7 und 4): Sind sie gültig? Wenn weiter mit 10, sonst mit 15. 10. Die Weiche öffnet sich und WP kommt bei LBE1 an und unterbricht diese. 11. FB1 teilt FB2 mit, dass ein WP weiter transportiert wird. 12. FB2 läuft und FB1 stoppt, wenn sich keine weiteren WPs drauf befinden. 13. Wiederhole Schritte 3-10 auf Anlage 2 14. FB2 stoppt und Roboter entnimmt das WP -> Ende 15. Weiche/Auswerfer schiebt WP in die Rutsche und LBC wird unterbrochen. 16. Wiederhole Schritte 1-9. |

|  |  |
| --- | --- |
| **Titel** | Prüfung des E-Stopp-Schalters |
| **ID** | U2 |
| **Akteure** | Kunde, WP, Anlage, E-Stopp-Schalter |
| **Ziel** | Es wird überprüft, ob alle Anlagen nach Betätigung des E-Stopp-Schalters ordnungsgemäß stillstehen und ob sie nach Betätigung der Reset-Taste an allen Anlagen wieder in den Betriebszustand versetzt werden können. |
| **Auslöser** | Der Kunde drückt den E-Stopp-Schalter |
| **Vorbedingung** | Alle Anlagen laufen fehlerfrei. |
| **Nachbedingung** | Alle Anlagen stehen still, bis der E-Stopp-Schalter herausgezogen und der Rest-Taster an allen Anlagen gedrückt wurde |
| **Erfolgsszenario** | 1. Kunde zieht zu einem beliebigen Zeitpunkt den E-Stopp-Schalter an einer beliebigen Anlage. 2. Die Anlage, deren Schalter gezogen wurde, informiert die andere Anlage über das Ereignis 3. Alle Anlagen kommen zum Still zustand 4. Alle grünen Lampen gehen aus und die roten Lampen fangen an zu blinken (Mit einer Frequenz von 1HZ) 5. Kunde zieht den E-Stopp-Schalter heraus und die Roten Lampen blinken nun mit einer Frequenz von 0,5 Hz 6. Der Kunde drückt den Reset-Taster an allen Anlagen und die roten Lampen erlöschen. 7. Anlagen wechseln in den Betriebszustand, dabei werden die Grünen Lampen wieder aktiviert. |

|  |  |
| --- | --- |
| **Titel** | WP kommt in die Rutsche |
| **ID** | U3 |
| **Akteure** | Kunde, WP, Anlage |
| **Ziel** | Es wird überprüft, ob ein defektes oder unzulässiges WP in die Rutsche kommt. |
| **Auslöser** | Der Kunde legt ein defektes oder unzulässiges WP bei LBB1 |
| **Vorbedingung** | Alle Anlagen laufen fehlerfrei und ein defektes oder unzulässiges WP wird auf dem FB1 gelegt. |
| **Nachbedingung** | Das WP kommt in die Rutsche |
| **Erfolgsszenario** | 1. Der Kunde legt ein defektes oder unzulässiges WP bei LBB1 2. Das WP wird bis HS1 transportiert. 3. Die Datenerfassung fängt an (und das System analysiert sie) 4. Das WP unterbricht LBH1 5. Das WP wird zu MS transportiert und MS prüft, ob das WP ein Metalleinsatz enthält 6. Das WP wird weiter zu LBS1 transportiert 7. Das WP kann zu keinem der vordefinierten Werkstückarten zugeordnet. 8. Die Weiche bleibt geschlossen und WP geht auf die Rutsche |

## Anlage: Analyse der technischen Gegebenheiten

### Technischer Aufbau und Hardwarekomponenten

Pro Anlage:

* FB: kann vor- und rückwärts fahren.
* Rutsche: beinhaltet Platz für max. 4 Werkstücke
* LEDs:
  + 2 LEDs (Q1, Q2) -> zur Signalisierung
  + 2 Taster-LEDs
  + 24 LEDs (auf dem Board) 12 sind für Output und 12 sind für Input gedacht, um die laufenden Funktionen der Anlange anzuzeigen
* Sensoren (5 LBs, HS, MS): zur Erfassung von Daten über die WKs (Position, Höhe, Metalleinsatz)
  + LBH: ist nicht zuverlässig (kann mit einem stück Plastik unterbrochen werden kann)

-> darf nicht benutzt werden

* + HS: hat Rauschen -> muss kalibriert werden
* Ampel (Rot, Gelb, Grün): zur Signalisierung von Zuständen/Warnungen. Mehrere Lampen können gleichzeitig leuchten.
* Tasten (Start, Stopp, Reset): zur Steuerung der Anlage -> s. R6
* E-Stopp: zum sofortigen Stoppen der Anlagen im Notfall.
* Aussortierer: dient der Aussortierung der WPs -> darf nicht minuten-lang aktiviert sein.

Ein Bild, das Diagramm, Plan, Design enthält.

Automatisch generierte Beschreibung

Abbildung 2: Skizze der Anlagen, zeigt die vorhandenen Sensoren und deren Platzierung/kürzel

## GPIO

Es gibt 32 Pins pro GPIO.

A close-up of a computer

Description automatically generated

Abbildung 3: GPIO-PINS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | LED | D | BIT | Funktion | HEX |
| GPIO:1 | OUT0 | 1 | 12 | Motor Rechtslauf | 0x0000\_1000 |
| OUT1 | 2 | 13 | Motor Linkslauf | 0x0000\_2000 |
| OUT2 | 3 | 14 | Motor langsam | 0x0000\_4000 |
| OUT3 | 4 | 15 | Motor Stop | 0x0000\_8000 |
| OUT4 | 5 | 16 | Rote Lampe an | 0x0001\_0000 |
| OUT5 | 6 | 17 | Gelbe Lampe an | 0x0002\_0000 |
| OUT6 | 7 | 18 | Grüne Lampe an | 0x0004\_0000 |
| OUT7 | 8 | 19 | Weiche öffnen | 0x0008\_0000 |
| GPIO:2 | OUT8 | 9 | 2 | LED Taste Start | 0x0000\_0004 |
| OUT9 | 10 | 3 | LED Taste Reset | 0x0000\_0008 |
| OUT10 | 11 | 4 | Signalleuchte Q1 | 0x0000\_0010 |
| OUT11 | 12 | 5 | Signalleuchte Q2 | 0x0000\_0020 |
| GPIO:0 | IN0 | 13 | 2 | WS im Einlauf // LBB | 0x0000\_0004 |
| IN1 | 14 | 3 | ~~WS in Höhenmessung~~ -> wird nicht benutzt | 0x0000\_0008 |
| IN2 | 15 | 4 | ~~WS Höhe Ok~~ -> wird nicht benutzt | 0x0000\_0010 |
| IN3 | 16 | 5 | WS in Weiche // LBS | 0x0000\_0020 |
| IN4 | 17 | 7 | WS Metall | 0x0000\_0080 |
| IN5 | 18 | 14 | Weiche offen | 0x0000\_4000 |
| IN6 | 19 | 15 | Rutsche voll // LBC | 0x0000\_8000 |
| IN7 | 20 | 20 | WS im Auslauf // LBE | 0x0010\_0000 |
| IN8 | 21 | 22 | TasteStart | 0x0040\_0000 |
| IN9 | 22 | 23 | TasteStop | 0x0080\_0000 |
| IN10 | 23 | 26 | TasteReset | 0x0400\_0000 |
| IN11 | 24 | 27 | E-Stop | 0x0800\_0000 |

### Werkstücke

Ein Bild, das Text, Screenshot, Kreis, Diagramm enthält.

Automatisch generierte Beschreibung

Abbildung 4: Richtige Werkstücke

Im Abbild sind die Werkstücke dargestellt, die von den Anlagen erwartet werden.

Falls andere Arten von Werkstücken aufgelegt werden, dann kann nicht garantiert werden, dass die Anlagen diese Arten erkennen und nach Anforderung sortieren können.

Das hohe Werkstück mit Metall darf nicht umgedreht (Metall nach unten) aufgelegt werden, da der Metallsensor in diesem Fall das Metall nicht erkennen kann.

## Softwareebene

### Schnittstellen: Nachrichten und Signale

|  |  |
| --- | --- |
| Trigger/Events von A1 | Bedeutung |
| M1\_LBB\_BLOCKED | Anlage 1: Lichtschranke am Anfang blockiert |
| M1\_LBB\_FREE | Anlage 1: Lichtschranke am Anfang frei |
| M1\_LBS\_BLOCKED | Anlage 1: Lichtschranke am Aussortierer blockiert |
| M1\_LBS\_FREE | Anlage 1: Lichtschranke an Aussortierer frei |
| M1\_LBC\_BLOCKED | Anlage 1: Lichtschranke an der Rutsche blockiert |
| M1\_LBC\_FREE | Anlage 1: Lichtschranke an der Rutsche frei |
| M1\_LBE\_BLOCKED | Anlage 1: Lichtschranke am Ende blockiert |
| M1\_LBE\_FREE | Anlage 1: Lichtschranke am Ende frei |
| M1\_START\_PRESSED | Anlage 1: Start gedrückt |
| M1\_START\_RELEASED | Anlage 1: Start nicht mehr gedrückt |
| M1\_STOP\_PRESSED | Anlage 1: Stop gedrückt |
| M1\_STOP\_RELEASED | Anlage 1: Stop nicht mehr gedrückt |
| M1\_RESET\_PRESSED | Anlage 1: Reset gedrückt |
| M1\_RESET\_RELEASED | Anlage 1: Zurücksetzen nicht mehr gedrückt |
| M1\_ESTOP\_PRESSED | Anlage 1: Not-Aus gedrückt |
| M1\_ESTOP\_RELEASED | Anlage 1: Not-Aus nicht mehr gedrückt |
| M1\_IN\_HIGHT\_MEASUREMENT | Anlage 1: In Höhenmessung |
| M1\_LEFT\_HIGHT\_MEASUREMENT | Anlage 1: Höhenmessung verlassen |
| M1\_METAL\_DETECTED | Anlage 1: Metall erkannt |

|  |  |
| --- | --- |
| Trigger/Events von A2 | Bedeutung |
| M2\_LBB\_BLOCKED | Anlage 2: Lichtschranke am Anfang blockiert |
| M2\_LBB\_FREE | Anlage 2: Lichtschranke am Anfang frei |
| M2\_LBS\_BLOCKED | Anlage 2: Lichtschranke am Aussortierer blockiert |
| M2\_LBS\_FREE | Anlage 2: Lichtschranke am Aussortierer frei |
| M2\_LBC\_BLOCKED | Anlage 2: Lichtschranke an der Rutsche blockiert |
| M2\_LBC\_FREE | Anlage 2: Lichtschranke an der Rutsche frei |
| M2\_LBE\_BLOCKED | Anlage 2: Lichtschranke am Ende blockiert |
| M2\_LBE\_FREE | Anlage 2: Lichtschranke am Ende frei |
| M2\_START\_PRESSED | Anlage 2: Start gedrückt |
| M2\_START\_RELEASED | Anlage 2: Start nicht mehr gedrückt |
| M2\_STOP\_PRESSED | Anlage 2: Stop gedrückt |
| M2\_STOP\_RELEASED | Anlage 2: Stop nicht mehr gedrückt |
| M2\_RESET\_PRESSED | Anlage 2: Reset gedrückt |
| M2\_RESET\_RELEASED | Anlage 2: Zurücksetzen nicht mehr gedrückt |
| M2\_ESTOP\_PRESSED | Anlage 2: Not-Aus gedrückt |
| M2\_ESTOP\_RELEASED | Anlage 2: Not-Aus nicht mehr gedrückt |
| M2\_IN\_HIGHT\_MEASUREMENT | Anlage 2: In Höhenmessung |
| M2\_LEFT\_HIGHT\_MEASUREMENT | Anlage 2: Höhenmessung verlassen |
| M2\_METAL\_DETECTED | Anlage 2: Metall erkannt |

Ein Bild, das Text, Diagramm, technische Zeichnung, Plan enthält.

Automatisch generierte Beschreibung

Abbildung 5: Sequenzdiagramm. Zeigt den Nachrichtenaustausch und den Ablauf, wie eine Entscheidung von FSMs getroffen und ausgeführt wird.

**E-Stop Funktionalität:**

Der E-Stop-Interrupt wird genauso wie andere Interrupts behandelt (siehe 5.1 Software-Architektur).

E-Stop-spezifische Behandlung:

* Der Encoder prüft den E-Stop-Pin als erstes. So wird die E-Stop-Pulse-Message als erstes an den Dispatcher geschickt.

Das hat zur Folge, dass die Anlagen darauf schneller reagieren.

Wenn der E-Stop gedrückt wird, stehen alle Anlagen still, bis der E-Stop herausgezogen wird und der Reset-Taster auf allen Anlagen gedrückt wird. Der Betrieb und die Reihenfolge der Werkstücke fangen dann wieder von neu an.

# Software-Design

## Software Architektur

Ein Bild, das Text, Diagramm, Screenshot, Plan enthält.

Automatisch generierte Beschreibung

Abbildung 6: Software-Architektur

Die Softwarearchitekturen SW1 und SW2 der Anlagen 1 und 2 sind nahezu identisch und unterscheiden sich lediglich in der Funktionsweise der FSMs. Beide Architekturen bestehen aus drei Hauptkomponenten: HAL, Logik und Kommunikation

- Die HAL kapselt die hardwarenahe Interaktion und beinhaltet die folgenden Klassen:

* Aktorik: Ermöglicht die Ansteuerung der Aktoren der jeweiligen Anlage (z.B. Förderband, Lampe).
* ISR: Bei Empfang eines Interrupts übermittelt die ISR den aktuellen Zustand des GPIO-0-Registers als Pulse-Nachricht an den Encoder.
* Encoder: Der Encoder analysiert diesen Zustand, identifiziert geänderte Pins und sendet für jeden der veränderten Pins eine separate Pulse-Nachricht an den Dispatcher. Diese Nachrichten enthalten sowohl die Pin-Nummer als auch die Art der Zustandsänderung (high-to-low oder low-to-high).

- Die Kommunikation ist für die interne und externe Kommunikation verantwortlich und umfasst:

* Dispatcher: Empfängt der Dispatcher eine Pulse-Nachricht vom Encoder, leitet er diese gezielt an die Komponenten weiter, die für die Verarbeitung solcher Nachrichten zuständig sind, wie beispielsweise Logik von SW1 oder Kommunikation von SW2.
* Receiver: Empfängt Signale vom Dispatcher der anderen Anlage und leitet diese an den Dispatcher seiner Anlage weiter.

- Die Systemlogik ist in mehreren Klassen unterteilt:

* FSMs (Finite State Machine): Die FSM-Klassen bilden die Zustandslogik des Systems ab. Sie definieren alle möglichen Zustände und steuern die Übergänge zwischen ihnen basierend auf Events, die vom Dispatcher empfangen werden.

- Der Remote-Controller verfügt über eine eigenständige Software, die es ihm ermöglicht, über die definierte Schnittstelle mit beiden Softwaresystemen zu kommunizieren.

# Implementierung: Besonderheiten

* Motor-Steuerung:
  + PINs:
    - BLOCK (Höchste Priorität)
    - MOVE\_RIGHT (2. Höchste Priorität)
    - SLOW (3. Höchste Priorität)
  + FSMs:
    - E-Stop, Fehler => benötigt Zugriff auf allen PINs
    - Begin => benötigt Zugriff auf PIN "MOVE\_RIGHT"
    - Hight Measurement => benötigt Zugriff auf PIN "SLOW"
    - Switch => steuert FB nicht
    - End => benötigt Zugriff auf PIN "BLOCK"
* Die PINs werden von genau einer FSM gesteuert. Da die PINs-Prioritäten, sowie sie per Default eingestellt sind, auch unserer Prioritäten entsprechen, sehen wir keine Probleme, die durch diesen Ansatz entstehen könnten.

Da die Anlagen nach einem E-Stop wieder von vorne anfangen müssen, stellt der Zugriff auf die PINs keine Probleme.

* Im Error Fall, muss das Band mit dem PIN "BLOCK" gestoppt werden. Da die FSMs darauf keinen Zugriff haben, melden sie den Fehler an eine "Fehler FSM". Die Fehler FSM hat zugriff auf den "BLOCK" PIN. Das kann zu Konflikten mit der E-Stop FSM führen, deshalb kann die HAL zwischen dem E-Stop-Block und dem Error-Block unterscheiden und kann sich entsprechend verhalten.
* Das Verschwinden eines WPs haben wir mit Hilfe von QNX-Timer umgesetzt. Jedes WP hat einen Timer, der am Anfang von jedem Bereich gestartet wird. Wenn diese Zeit abläuft, bekommt das System ein Interrupt mit dem Code „PULSE\_TIME\_OUT\_MAX“. Dies führt dann zu einem Fehler und alle FSMs warten auf „ERROR\_OK“. Wenn das WP beim nächsten Sensor ankommt, bevor der Timer abläuft, dann wird der Timer gelöscht.
* Das Auftauchen eines WPs wird mithilfe von Time Stamps umgesetzt. Wenn ein WP bei LBB aufgelegt wird, dann wird ein Time Stamp gespeichert. Erreicht dieses WP dann den nächsten Sensor, wird dann ein weiterer Time Stamp gemacht. Wenn die Zeit, die seit dem letzten Time Stamp vergangen ist, mit der vordefinierten Zeit übereinstimmt, dann gibt es keinen Fehler. Wenn nicht, wechseln alle FSMs in einen Fehlerzustand.
* Beim Übergang zwischen Anlage 1 und Anlage 2:
  + Nachdem Anlage 1 ein WP geschickt hat, wartet sie auf ein ACK. Wenn das ACK nicht kommt dann gibt es erstmal keinen Fehler. Erst wenn ein neues WP geschickt werden muss und das ACK ist nicht angekommen, dann gibt es einen Fehler.
  + Wenn das WP zwischen Anlage1 und Anlage2 verschwindet, dann kommt es zu einem Error nur auf Anlage2. Aber dieser Error heißt, dass Anlage1 den ACK nicht bekommt und dass es später zu einem Error kommt, außer man quittiert den Fehler auf Anlage2 bevor ein weiteres WP transportiert werden muss.

# Qualitätssicherung

## Teststrategie

* Es wurden nur manuelle Tests an den Anlagen durchgeführt.

## Testszenarien/Abnahmetest

Es gilt für alle Test die Folgende Vorbedingung (es sei denn, spezifische Änderungen sind angegeben.):

Am Anfang des FB1 wird jeweils ein WP platziert und losgelassen, LBB1 wird unterbrochen. Die Anlagen sind Verbunden, richtig kalibriert und im Betrieb-Modus. Die Identifizierung der WP funktioniert problemlos. Mindestabstand wird eingehalten.

|  |  |
| --- | --- |
| **T\_01** | Test des Sortier-Verfahrens (bei korrekter Reihenfolge). |
| Requirements: | R1.1, R1.2, R1.7, R2.1, R2.2, R2.3, R2.4, R3.1, R4.2, R4.5 |
| Vorbedingungen: |  |
| Testsequenz | WPs auflegen -> WPs werden identifiziert -> die WPs werden durchgelassen. |
| Input | Die WPs sollen am Anfang von FB1 platziert werden, in folgender Reihenfolge:  1.WP mit Bohrung mit Metall  2.WP mit Bohrung mit Metall  3.Flaches WP |
| Output | Am Ende von FB2 werden die WPs in folgender Reihenfolge ankommen:  1.WP mit Bohrung mit Metall  2.WP mit Bohrung mit Metall  3.Flaches WP |

|  |  |
| --- | --- |
| **T\_02** | WP nicht in der korrekten Reihenfolge platziert |
| Requirements: | R1.1, R1.3, R1.7, R2.1, R2.2, R2.4, R2.5, R3.1, R4.2 |
| Vorbedingungen: | Die Rutsche1 hat noch freie Plätze. |
| Testsequenz | Testsequenz für das fehlerhafte WP:  Fehlerhaftes WP auflegen -> WP wird identifiziert -> das WP fällt in die Rutsche1. |
| Input | Die WPs sollen am Anfang von FB1 platziert werden, in folgender Reihenfolge:  1.WP mit Bohrung mit Metall  2.WP mit Bohrung mit Metall  3.Beliebiges WP (aber nicht flaches WP) oder ein defektes WP  4.Flaches WP |
| Output | Am Ende von FB2 werden die WPs in folgender Reihenfolge ankommen:  1.WP mit Bohrung mit Metall  2.WP mit Bohrung mit Metall  3.Flaches WP |

|  |  |
| --- | --- |
| **T\_03** | Test der Erkennung von aufgetauchten WPs |
| Requirements: | R1.6, R2.2, R2.5, R3.1, R3.2, R5.3 |
| Vorbedingungen: | LBB1 wird nicht unterbrochen. |
| Testsequenz | WP auflegen -> WP löst ein Sensor aus -> Ein Error wird gemeldet. |
| Input | Ein WP auflegt, ohne dass die LBB1 unterbrochen wird. (oder auflegen, wo/wann es nicht erwartet wird) |
| Output | Error Meldung (Band-Stop, die rote Lampe blinkt) |

|  |  |
| --- | --- |
| **T\_04** | Test der Erkennung von verschwundenen WPs |
| Requirements: | R5.2 |
| Vorbedingungen: | Ein richtig-platziertes WP (die Position ist irrelevant) wird beim nächsten Sensor (Lichtschranke oder HS) erwartet. |
| Testsequenz | WP wird erwartet -> WP wird vom Band entnommen -> Ein Error wird gemeldet. |
| Input | Das WP vom Band entnehmen. |
| Output | Error Meldung (Band-Stop, die rote Lampe blinkt) |

|  |  |
| --- | --- |
| **T\_05** | Test des Durchlassens bei voller Rutsche |
| Requirements: | R1.4, R5.1 |
| Vorbedingungen: | Ein falsches WP soll demnächst aussortiert werden. Die Rutsche von Anlage1 ist voll bzw. blockiert. |
| Testsequenz | WP kommt beim Aussortierer von Anlage1 an -> WP wird durchgelassen. |
| Input | Ein beliebiges WP |
| Output | Das WP wird durchgelassen |

|  |  |
| --- | --- |
| **T\_06** | Test des Identifizierungsverfahrens. |
| Requirements: | R1.1, R1.6 |
| Vorbedingungen: | Ein WP wird demnächst unter dem HS ankommen. |
| Testsequenz | WS kommt unter den HS an -> FB fährt langsamer -> WP wird gemessen -> FB fährt schneller (WP verlässt den HS-Bereich) |
| Input | Ein beliebiges WP |
| Output | Der Typ des WPs wird auf der Konsole ausgegeben. |

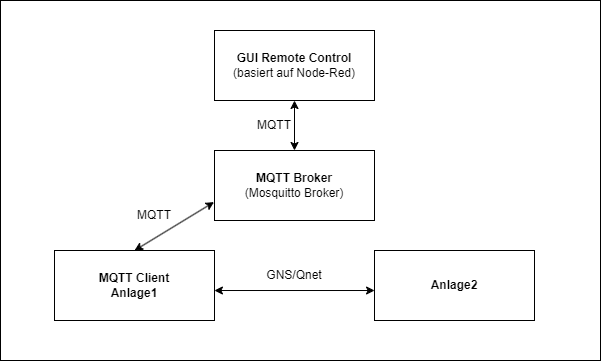
|  |  |
| --- | --- |
| **T\_07** | Test des Sortierungsverfahrens (bei falscher Reihenfolge). |
| Requirements: | R1.3, R1.5 |
| Vorbedingungen: | Die Rutsche1 ist leer. |
| Testsequenz | WPs auflegen -> WPs werden identifiziert -> die WPs werden aussortiert. |
| Input | Die WPs sollen am Anfang von FB1 platziert werde (Reihenfolge ist irrelevant):  1) Flaches WP. 2) Binäres WP. 3) Hohes WP mit Bohrung ohne Metalleinsatz. 4) Hohes WP ohne Bohrung (umgedreht) ohne Metalleinsatz |
| Output | Die aufgelegten WPs landen in die Rutsche1 in derselben Reihenfolge, in der sie aufgelegt werden. |

|  |  |
| --- | --- |
| **T\_08** | Test der Erkennung, WP kommt zum falschen Zeitpunkt (zu früh/zu spät) |
| Requirements: | R1.1, |
| Vorbedingungen: | Ein WP wird zu einem Bestimmten Zeitpunkt (Range) bei einem Sensor erwartet. |
| Testsequenz | WP löst ein Sensor zu einem falschen Zeitpunkt aus -> Ein Error wird gemeldet |
| Input | Ein WP daran verhindern zu einem richtigen Zeitpunkt anzukommen (AKA WP aufhalten oder nach vorne schieben). |
| Output | Error Meldung (Band-Stop, die rote Lampe blinkt) |

# Technische Schulden

* Weiche/Auswerfer Unterscheidung bei einem „Warmen“ Start.
* Wenn die Rutsche beim Start des Systems bereits voll ist, wird dies nicht durch die Software erkannt.
* Wenn E-Stop beim Start des Systems bereits gedrückt ist, wird dies nicht durch die Software erkannt.
* Auf die Platzierung des WS wird nicht beachtet. Es wird davon ausgegangen, dass der User das WP auf die „Optimale Stelle“ (möglichst links in Fahrtrichtung) platziert.
* Verbindungsabriss führt dazu, dass die Anlagen neu gestartet und kalibriert werden müssen.
* Wenn das Band geblockt wird während die Weiche sich öffnet, kann es sein, dass es zu einem Fehler kommt.
* Wenn man zu oft die Tasten Start und Stop drückt, dann ist es möglich, dass die Analgen in den Fehlerzustand gehen.
* Wenn das WP M1\_LBE verlässt, aber M2\_LBB noch nicht erreicht hat und die Stop Taste gedrückt wird, dann zählt das WP als verschwunden.
* Wenn ein WP am Ende von Anlage 1 (LBE ist blockiert) wartet bis Anlage 2 frei wird, dann fährt FB1 weiter, wenn Anlage 2 frei wird, unabhängig davon, ob Stop-Taste auf Anlage 1 gedrückt ist oder nicht.
* Nach einem Verbindungsabriss: Estop auf einer Anlage drücken -> Estop herausziehen -> reset drücken auf der selben Anlage -> die Verbindung wieder herstellen -> dann kann es sein dass Estop auf dieser Anlage quittiert wird aber auf der anderen Anlage nicht.
* Wenn die Stop-Taste gedrückt wird, dann dauert es ein paar Millisekunden, bis das Band stoppt. Wenn ein WP einen Sensor innerhalb dieser Zeit auslöst, dann geht die Anlage in Fehlerzustand.
* Wenn ein Fehlerauftritt, dann müssen alle WPs, die auf dem Band sind entfernt werden:
* Idee, wie dieses Problem behoben werden könnte:
  + WPs, die einen Fehler verursacht haben, werden als nicht mehr auf dem Band markiert. Und es Wird davon ausgegangen, dass sie entfernt worden.
  + Die Fsms counter müssen verändert werden. Wenn ein trigger kommt, dann sucht die FSM nach dem ersten „gültigen“ WP im Vector. „Gültig“ heißt zb. Für HightMeasurementFSM das erste WP, das noch nicht unter em HS war bzw. einen Typ zugewiesen wurde.
  + Bei Error: bekommen die anderen FSMs das nicht mit, sodass sie nicht unterbrochen werden, falls sie zum Zeitpunkt des Errors ein WP behandeln.
  + Da die FSMs nicht mitbekommen, dass ein Fehler passiert ist, kann es sein, dass sie das Band unblockieren. Um das zu lösen, muss die HAL schlauer gemacht werden.
* Ungefähr schätzen wir, dass diese Lösung 50 Stunden dauert (mit Anlagen).

# Remote Control:



Ziel:

Die Anlagen sollen von einem Remote-System gesteuert und überwacht werden. Die Kommunikation soll auf MQTT basieren und die Remote-Control-Anwendung soll eine graphische Oberfläche haben. Diese Oberfläche soll als Web-Anwendung realisiert werden.

Anforderungen:

1. MQTT-Kommunikation:

Die Kommunikation zwischen den Anlagen und dem Remote-System erfolgt über MQTT.

1. Graphische Benutzeroberfläche:

Die Remote-Control-Anwendung soll eine graphische Oberfläche haben, als Web-Anwendung implementiert wird.

1. Button-Panel:

Die Button-Panel der Anlagen sollen in der Oberfläche sichtbar und benutzbar sein.

1. Anzeige relevanter Informationen:

Dazu gehört z.B., dass eine Rutsche voll ist. Dies soll graphisch dargestellt werden.

1. Konsolenausgaben:

Die Ausgaben auf der Konsole der Anlagen soll auch in der Remote-Control-Anwendung ausgegeben werden.

Schritte zur Implementierung:

1. Einrichtung des MQTT-Brokers.
2. Entwicklung der Web-basierten Remote-Control-Anwendung mit Node-Red.

In dem Projekt wurde einen MQTT-Client implementiert, der sich mit einem auf dem Rechner laufenden Mosquitto-Broker verbindet.

**Die Hauptbesonderheiten dieser Implementierung sind:**

**Client-Definition und Verbindungsaufbau:**

Der Client wird mit der Broker-Adresse tcp://ipAdresse von Anlage1:1883 (z.B. tcp://192.168.101.4:1883 verbunden).

Der Verbindungsaufbau erfolgt durch die connectClient-Funktion, die auch Callback-Funktionen für Nachrichtenempfang (msgarrvd), Verbindungsverlust (connlost) und Bestätigung der Nachrichtenlieferung (delivered) setzt.

**Topics und Nachrichten:**

Der Client abonniert Topics wie "ampeln/anlage1", "buttons/anlage1", etc.

Nachrichten werden über die publishMsg-Funktion veröffentlicht, z.B. auf Topics wie "console", "chute1", "chute2".

Der Nachrichtenempfang wird in msgarrvd verarbeitet, wo je nach empfangenem Wert verschiedene Aktionen über Pulsnachrichten (MsgSendPulse) ausgelöst werden.

**Integration mit Node-RED:**

Eine grafische Oberfläche basiert auf Node-RED, wo alle Nodes über mqtt-in und mqtt-out mit dem MQTT-Broker verbunden sind.

Die MQTT-Nodes in Node-RED verwenden die gleiche Broker-Adresse und ermöglichen die visuelle Steuerung und Überwachung der Anlage.

# Betriebsanleitung

## Kalibrierungsmodus:

Bevor die Anlage in den Betriebsmodus gehen darf, muss eine Kalibrierung durchgeführt werden. Ohne diese Kalibrierung wird die Software nicht richtig funktionieren. Um in den Kalibrierungsmodus zu gelangen, muss die Start-Taste mindestens eine Sekunde lang gedrückt werden. Sobald die grüne Lampe anfängt zu blinken, kann mit der Kalibrierung begonnen werden. Es ist zu beachten, dass die Rutsche mindestens einen freien Platz haben muss, bevor die Kalibrierung gestartet wird.

Für die Kalibrierung dürfen nur flache Werkstücke verwendet werden. Legen Sie ein flaches Werkstück an den Anfang der ersten Lichtschranke und achten Sie darauf, dass es so weit wie möglich links auf dem Band positioniert ist. Das Band wird sich dann schnell bewegen und das Werkstück wird aussortiert.

Sobald das Band stoppt, kann ein weiteres flaches Werkstück gelegt werden. Dieses Werkstück muss das Ende des Bandes erreichen, woraufhin das Band stoppt. Die Kalibrierung ist abgeschlossen, wenn die grüne Lampe aufhört, zu blinken. Danach können Sie durch erneutes Drücken der Start-Taste in den Betriebsmodus wechseln.

Während der Kalibrierung darf sich maximal ein Werkstück gleichzeitig auf dem Band befinden. Es ist wichtig, die genannten Regeln genau zu befolgen, um Softwarefehler zu minimieren. Sollte ein Fehler während der Kalibrierung auftreten, muss der Code neugestartet werden.

* Die erfassten Daten während der Kalibrierung:
  + - Die maximale und minimale Höhe des FBs.
    - Die maximale und minimale Höhe eines Flachen WPs.
    - Die Längen der Sections (A,B,C,D) des FBs in Zeit (beim schnellen Motorlauf).

## Betriebsmodus:

In den Betriebsmodus gelangt man, wenn man die Start-Taste weniger als eine Sekunde drückt. Wenn die grüne Lampe anfängt zu leuchten, können Werkstücke am Anfang des Bandes gelegt werden. Dabei ist der Mindestabstand einzuhalten. Den Mindestabstand erkennt man an der Q1-LED. Wenn diese leuchtet, dann darf ein neues Werkstück aufgelegt werden. Wenn E-Stop gedrückt wird, dann leuchtet die Q2-LED an der Anlage, an der E-Stop gedrückt wurde. Die Bedeutung der anderen Tasten bzw. LEDs sind unter dem Kapitel 3.2.1 zu finden.

# Modellierung/FSMs Konzept

Jede Anlage hat im Betrieb Modus FSMs, die einen Bereich von der gesamten Anlage Modellieren oder eine bestimmte separate Funktionalität erfüllen. Z.B. “BeginFSM” modelliert LBB und ist für den Mindestabstand zuständig, “ErrorFSM” ist für die Behandlung von Errors zuständig.

Auf jeder Anlage gibt es insgesamt 8 FSMs:

* EStopFSM: immer aktiv, zuständig für die Behandlung von Estop. Im Fall von Estop, teilt sie anderen FSMs mit, wann sie wieder aktiv sein dürfen.
* ModesFSM: zuständig für die Kalibrierung. Sie teilt anderen FSMs mit, wenn sie in Operation geht. Steuert die grüne Lampe.
* StopFSM: Behandlung von Stop. resetet/restartet Timer entsprechend.
* ErrorFSM: Behandlung von Errors. Wenn ein Error auftritt, dann wird alles, bis auf Reihenfolge, geresettet inkl. Data. D.h. alles was auf dem FB ist, muss entfernt werden. Steuert die Rote Lampe.
* BeginFSM: verwendet LBB. Zuständig für den Mindestabstand.
* HightMeasurementFSM: Modelliert den Höhensensor. Ist dafür zuständig, dass das FB langsam und wieder schnell fährt.
* SwitchFSM: verwendet den Metallsensor, Aussortierer und LBC.
* EndFSM: verwendet LBE und ist für den Übergang zu Analge2 zuständig.

Die FSMs sind orthogonal und laufen in einem Thread. Der FSM-Thread bekommt über den Kernelaufruf „MsgReceivePulse“ Pulse-Codes und verteilt diese an die FSMs, die sich dafür interessieren.

Die FSMs haben einen Zugriff auf eine gemeinsame Context-Data Klasse. In dieser Klasse werden Daten, die von mehreren FSMs benötigt werden, gespeichert, wie z.B. eine Queue von WP-Objekten.

Jedes WP-Objekt bekommt bei M1\_LBB eine eigne ID zugewiesen. Der Typ des WPs wird beim Höhensensor festgelegt „Flat“ oder „Defective“ und wenn beim Metallsensor Metall erkannt wird, dann wird der Typ zu „Metal“ geändert.

Die Typerkennung der Werkstücke ist so definiert, dass während der Kalibrierung ein flaches WP aufgelegt wird und damit dann die maximale Höhe des WPs bestimmt wird. Diese Höhe wird dann für den Service-Modus verwendet, um den Typ des WPs festzustellen.

# Lessons Learned

* Nicht viel Zeit mit der Modellierung verschwenden.
* Frühzeitig mit der Programmierung anfangen, sonst bleibt am Ende nicht viel Zeit übrig dafür.
* In der Modellierung kann man viele Fälle übersehen
* Am besten implementiert man eine FSMs bevor man mit der Modellierung komplett fertig ist. Die Erfahrung bei der Implementierung hilft bei der Modellierung.
* Es ist eine gute Idee, zentralisiertes verhalten in einer separaten FSM zu modellieren. Zb. ErrorFSM. Auch wenn das Verhalten von Unterschiedlichen Events und in Verschiedenen Kontexten verursacht wird. FSMs können miteinander interagieren.

# Anhang

## Abkürzungen

* FB:
  + FB1 -> Förderband der ersten Anlage
  + FB2 -> Förderband von der zweiten Anlage
* HS:
  + HS1 -> Höhensensor der ersten Anlage
  + HS2 -> Höhensensor der zweiten Anlage
* LB:
  + LB -> Lichtschranke (Lightbarrier)
  + LBB: (Begin)
    - LBB1 -> LB am Anfang der ersten Anlage
    - LBB2 -> LB am Anfang der zweiten Anlage
  + LBH:
    - LBH1 -> LB beim HS der ersten Anlage
    - LBH2 -> LB beim HS der ersten Anlage
  + LBS: (Switch)
    - LBS1 -> LB bei der Weiche der ersten Anlage
    - LBS2 -> LB bei der Weiche der zweiten Anlage
  + LBC: (Chute)
    - LBC1 -> LB bei der Rutsche der ersten Anlage
    - LBC2-> LB bei der Rutsche der zweiten Anlage
  + LBE: (End)
    - LBE1 -> LB am Ende der ersten Anlage
    - LBE2 -> LB am Ende der zweiten Anlage
* MS:
  + MS1 -> Metallsensor der ersten Anlage
  + MS2 -> Metallsensor der zweiten Anlage
* WP/WPs -> Werkstück/Werkstücke (Workpiece)

## Fragen, die Während des Projekts aufkamen

* Was ist damit gemeint? „Das gleiche Namensschema mit dem Postfix ”\_RDD” gilt für den Dateinamen von Ihrem RDD.“ (S. 8)
  + Am Ende jeder Datei den Präfix „\_1\_2“ schreiben (für Praktikumsgruppe 1 Team 2)
* Es gibt keine User Stories -> wie Stakeholder?
  + Profs, Studenten, HAW
* Lampe: was sind Warnungen?
  + kein Eingriff vom Bediener notwendig, z.B.: volle Rutsche führt zu einer Warnung
* was ist Service Mode?
  + Selbsttest, Kalibrierungsmodus
* Was passiert nach dem die korrekte Reihenfolge einmal durchgelassen wurde?
  + Die Anlangen arbeiten weiter und die Reihenfolge wiederholt sich
* Wenn beide Rutschen voll sind und ein defektes WP erkannt wurde, soll der Bandstopp dann erfolgen, wenn das WP auf FB1 oder FB2 ist?
  + Auf FB2, da in der Zwischenzeit ein WP von Rutsche2 entfernt werden könnte und somit wieder Platz für das Aussortieren des defekten WP frei wäre.
* Was passiert, wenn die Anlagen fehlerfrei laufen und die Reset-Taste gedrückt wurde?
  + Nichts -> Reset-Taste ist nur bei Fehlerzuständen notwendig um einen Fehler zu quittieren.
* Was passiert, wenn die Anlagen wieder in den Betriebszustand wechseln, nachdem die Stop-Taste gedrückt wurde? Sollen sie weiter machen, wo sie aufgehört haben oder von neu anfangen?
  + Stop entspricht semantisch einer Pause -> nur an der jeweiligen Anlage wird pausiert, solange nichts Katastrophales (E-Stop) passiert -> Wenn danach wieder Start gedrückt wird, dann läuft die Anlage weiter.
  + E-Stop -> Zustände werden zurückgesetzt, WP werden entnommen. Reihenfolge wird neu anfangen -> Als hätte man die Anlagen komplett neu gestartet
* Was ist der Ruhezustand?
  + Anlagen sind an, aber Start-Taste wurde noch nicht gedrückt (Punkt Stop-Taste ist zu berücksichtigen).