

Dieses RDD Template stellt eine grobe Struktur des Dokuments dar.

Sie können es (fast) nach Belieben verändern.

Der rot-kursive Text soll durch Ihren Text ersetzt werden.

Bitte aktualisieren Sie bei jeder Meilenstein-Abnahme das Inhaltsverzeichnis.

# Requirements and Design Documentation (RDD)

SE2P – Praktikum – Wintersemester 2014/2015

Kirstein, Katja, 2125137, katja.kirstein@haw-hamburg.de

Kowalka, Anne-Lena, 2081899, anne-lena.kowalka@haw-hamburg.de

Triebe, Marian, 2124897, marian.triebe@haw-hamburg.de

Winter, Eugen, 2081992, eugen.winter@haw-hamburg.de

## Änderungshistorie:

Version	Autor	Datum	Anmerkungen/Änderungen
0.1	Thomas Lehmann	17.09.2014	Überarbeitung des Templates
0.2	Anne-Lena Kowalka	07.10.2014	Use Cases, Use Case Diagramm, Requirements und Arbeitspakete hinzugefügt
0.3	Eugen Winter	16.10.2014	Abnahmetests hinzugefügt

## Inhaltsverzeichnis

Motivation .....	4
Teamorganisation .....	4
1.1 Verantwortlichkeiten .....	4
1.2 Absprachen .....	4
Projektplan .....	4
1.3 PSP/Zeitplan/Tracking .....	4
1.4 Repository-Konzept .....	5
1.5 Qualitätssicherung .....	5
Randbedingungen .....	5
1.6 Entwicklungsumgebung .....	5
1.7 Werkzeuge .....	5
1.8 Sprachen .....	5
Requirements und Use Cases .....	5
1.9 Stakeholder .....	5
1.10 Anforderungen (Requirements) .....	6
1.10.1 Functional Requirements .....	6
1.10.2 Non-functional Requirements .....	10
1.11 Use Cases .....	11
1.11.1 ID: UC1 .....	11
1.11.2 ID: UC2 .....	12
1.11.3 ID: UC3 .....	12
1.11.4 ID: UC4 .....	13
1.11.5 ID: UC5 .....	13
1.11.6 ID: UC6 .....	14
1.11.7 ID: UC7 .....	14
1.11.8 ID: UC8 .....	15
1.12 Use-Case-Diagramm .....	16
1.13 Systemanalyse .....	17
Design .....	17
1.14 System Architektur .....	17
1.15 Datenmodell .....	17
1.16 Verhaltensmodell .....	17

Implementierung.....	17
1.17    Algorithmen.....	17
1.18    Patterns .....	17
1.19    Mapping Rules .....	17
Testen.....	17
1.20    Unit Test/Komponenten Test .....	18
1.21    Integration Test/System Test .....	18
1.22    Regressionstest .....	18
1.23    Abnahmetest .....	18
1.24    Testplan.....	18
1.25    Testprotokolle und Auswertungen .....	18
Lessons Learned .....	18
Glossar .....	19
Abkürzungen.....	19
Anhänge.....	19
1.26    Coding Style: Google C++.....	19
1.26.1    General rules .....	19
1.26.2    Naming.....	20
1.26.3    Headers.....	21
1.26.4    Breaking Statements.....	21

## Motivation

Es gilt, eine Werkstücksortieranlage zu programmieren. Die Anlage besteht aus zwei Förderbändern, die durch eine serielle Schnittstelle miteinander verbunden sind und jeweils durch einen eigenen GEME-Rechner gesteuert werden.

Es gibt drei Werkstücke unterschiedlicher Art (flach, mit Bohrung und Metall, mit Bohrung ohne Metall). Am Ende von Band 2 sollen nur normal hohe Werkstücke mit Bohrung nach oben ankommen, wobei sich Werkstücke mit bzw. ohne Metall abwechseln.

## Teamorganisation

Gibt es wichtige Regeln oder Anmerkungen zur Teamorganisation?

### 1.1 Verantwortlichkeiten

Entscheidungen werden gemeinsam im Team gefällt.

Ansprechpartner gesamtes Team: Eugen Winter

Github-Verwaltung: Marian Triebe

Protokollführerin: Anne-Lena Kowalka

RDD-Führerin: Anne-Lena Kowalka

Implementierung: Katja Kirstein, Anne-Lena Kowalka, Marian Triebe, Eugen Winter

Testen: Katja Kirstein, Anne-Lena Kowalka, Marian Triebe, Eugen Winter

### 1.2 Absprachen

Termin für Meetings: Freitags 14:00 (Stand: 26.9.2014)

Weitere Termine nach Absprache

## Projektplan

In diesem Kapitel sollten organisatorische Punkte beschrieben und festgelegt werden.

### 1.3 PSP/Zeitplan/Tracking

Verwendetes Vorgangsmodell: V-Modell

Festgestellte Arbeitspakete mit ihrer zugehörigen Dauer (Stand: 7.10.2014) : Requirements feststellen (5h gesamt), Use Cases feststellen (5h gesamt), RDD bearbeiten (2h/Woche), Git-Repository-Verwaltung (1h/Woche), Codequalität sicherstellen (1h/Woche), Schnittstellenansteuerung/Interface (3h gesamt), Projektplanung mit GANTT (3h gesamt), Regressionstests (8h gesamt), Protokollführung (1h/Woche), Meetings abhalten (1,5h/Woche), Moderation der Meetings/ Agenda erstellen (30 Min/Meeting bzw. Woche), UML-Diagramme erstellen (8h gesamt), Implementierung → unterteilt sich in: HAL bzw. Ports ansteuern (→ Tasten, Lichter, serielle Schnittstelle, Sensorik) (20h gesamt), Tests (16h), Fehlerbehandlung/-korrektur (30h gesamt)

Projektstrukturplan, Ressourcenplan, Zeitplan, Abhängigkeiten von Arbeitspaketen, Eventueller Zeitverzug, etc.

#### 1.4 Repository-Konzept

Coding Style: Google C++ Style Guide, konkrete Beschreibung im Anhang

Branching Strategie: Auf dem *master* Branch befindet sich nur der aktuelle Milestone. Aktuelle Entwicklungen finden auf dem *develop* Branch statt.

Es gibt die Möglichkeit Features-Branches auf Basis des *develop* Branches zu erstellen.

#### 1.5 Qualitätssicherung

Durch Testen nach jeder Phase bzw. Unit-Tests soll die Qualität sichergestellt werden.

Wie wollen Sie die Qualität in ihrem Projekt sicherstellen?

### Randbedingungen

#### 1.6 Entwicklungsumgebung

Betriebssystem: QNX, Simulator, Geme Computer

#### 1.7 Werkzeuge

Momentics IDE für QNX, Doxygen, eventuell Cpp-Unit

#### 1.8 Sprachen

C++ (nur STL)

### Requirements und Use Cases

#### 1.9 Stakeholder

Entwickler, Tester, Kunde, Betreuer, Personal

## 1.10 Anforderungen (Requirements)

### 1.10.1 Functional Requirements

<u>ID</u>	<u>Titel</u>	<u>Bereich</u>	<u>Beschreibung</u>
FR-001	Typen von Werkstücken	Werkstück	Es gibt 3 Typen von Werkstücken: Zu flach, mit Bohrung, sowie mit und ohne Metalleinsatz.
FR-002	Anzahl der Sortierbänder	Anlage	Für die Werkstück-Sortieranlage stehen 2 Sortierbänder zur Verfügung.
FR-003.1	Ziel der Werkstück-Sortieranlage	Anlage	Am Ende von Band 2 sollen die Werkstücke im Wechsel mit und ohne Metalleinsatz ankommen.
FR-003.2	Ziel der Werkstück-Sortieranlage	Anlage	Am Ende von Band 2 sollen die einzelnen Werkstücke mit der Bohrung nach oben liegend ankommen.
FR-004	Zuführung eines Werkstücks in die Werkstück-Sortieranlage	Anlage	Das Werkstück wird vom Personal an den Anfang von Band 1 in den Bereich der Lichtschranke gelegt.
FR-005	Entnahme eines Werkstücks aus der Werkstück-Sortieranlage	Anlage	Nach dem erfolgreichen Durchlaufen der Sortieranlage, wird das Werkstück am Ende von Band 2 vom Personal entnommen.
FR-006.1	Erkennen und Aussortieren von zu flachen Werkstücken	Sensorik	Zu flache Werkstücke werden auf Band 1 mit Hilfe der Höhenmessung erkannt.
FR-006.2	Erkennen und Aussortieren von zu flachen Werkstücken	Anlage	Nach der Erkennung werden zu flache Werkstücke auf Band 1 mit Hilfe der Weiche aussortiert.
FR-007.1	Erkennen und Ausrichten verkehrt liegender Werkstücke	Sensorik	Mit Hilfe der Höhenmessung erkennt Band 1, ob ein Werkstück mit der Bohrung nach oben oder unten auf das Band gelegt wurde.
FR-007.2	Erkennen und Ausrichten verkehrt liegender Werkstücke	Anlage	Das verkehrt liegende Werkstück mit der Bohrung nach unten wird an das Ende von Band 1 befördert und die Anlage hält an.
FR-007.3	Erkennen und Ausrichten verkehrt liegender Werkstücke	Anzeige	Die gelbe Signalleuchte der Ampelanlage von Band 1 signalisiert mit einem Blinken dem Personal, dass ein Wenden des Werkstücks notwendig ist.
FR-007.4	Erkennen und Ausrichten verkehrt liegender Werkstücke	Sensorik	Ein Timer wird beim Eintreten in den wartenden Zustand gestartet und räumt dem Personal zum Wenden des Werkstücks eine vordefinierte Zeitspanne ein.
<u>ID</u>	<u>Titel</u>	<u>Bereich</u>	<u>Beschreibung</u>
FR-007.5	Erkennen und Ausrichten verkehrt liegender Werkstücke	Personal	Das Personal wendet das Werkstück von Hand mit der Bohrung nach oben, quittiert den Hinweis und startet die Anlage wieder.

FR-007.6	Erkennen und Ausrichten verkehrt liegender Werkstücke	Fehler	Es kommt zu einer Fehlermeldung, wenn das Werkstück nicht innerhalb der vordefinierten Zeitspanne wieder zurück auf das Ende von Band 1 gelegt worden ist.
FR-008.1	Erkennen und Aussortieren verkehrt liegender Werkstücke	Sensorik	Mit Hilfe der Höhenmessung erkennt Band 2, ob ein Werkstück mit der Bohrung nach unten auf dem Band liegt.
FR-008.2	Erkennen und Aussortieren verkehrt liegender Werkstücke	Anlage	Nach Erkennung eines verkehrt liegenden Werkstücks auf Band 2, wird dieses mit Hilfe der Weiche aussortiert.
FR-009.1	Einhalten der korrekten Reihenfolge der Werkstücke	Sensorik	Mit Hilfe des Metallsensors auf Band 2 wird erkannt, ob hintereinander zwei gleichartige Werkstücke (mit/ohne Metalleinsatz) befördert worden sind.
FR-009.2	Einhalten der korrekten Reihenfolge der Werkstücke	Anlage	Nach Erkennung der falschen Reihenfolge wird das betroffene Werkstück an den Anfang von Band 2 befördert.
FR-009.3	Einhalten der korrekten Reihenfolge der Werkstücke	Anzeige	Die gelbe Signalleuchte der Ampelanlage von Band 2 signalisiert mit einem Blinken dem Personal, dass ein Entfernen des Werkstücks notwendig ist.
FR-009.4	Einhalten der korrekten Reihenfolge der Werkstücke	Personal	Das Personal entfernt das betroffene Werkstück von Hand, quittiert den Hinweis und startet die Anlage erneut.
FR-010	Hinzufügen von Werkstücken auf Anfang von Band 1	Anlage	Es dürfen, sobald der Anfang mit der Lichtschranke von Band 1 frei ist, weitere Werkstücke auf den Anfang von Band 1 gelegt werden.
FR-011	Anzahl der Werkstücke auf Band 1	Anlage	Es dürfen sich mehrere Werkstücke zeitgleich auf Band 1 befinden.
FR-012.1	Übergabe von Werkstücken von Band 1 an Band 2	Anlage	Die Werkstücke von Band 1 werden einzeln an Band 2 übergeben, wenn dieses frei ist.
FR-012.2	Übergabe von Werkstücken von Band 1 an Band 2	Anlage	Es darf sich nur ein Werkstück zeitgleich auf Band 2 befinden.
FR-013	Identifizieren eines Werkstücks	Werkstück	Jedes Werkstück bekommt intern nach der Vermessung auf Band 1 eindeutige Werkstück- Eigenschaften zugewiesen.
FR-014	Zusammensetzung der Werkstück-Eigenschaften	Anlage	Die Werkstück-Eigenschaften beinhalten: ID aus fortlaufender Zahl, Typ des Werkstücks (zu flach, mit Metalleinsatz, ohne Metalleinsatz, Bohrung nach oben, Bohrung nach unten) und den Höhenmesswerten von Band 1 und Band 2.

<b><u>ID</u></b>	<b><u>Titel</u></b>	<b><u>Bereich</u></b>	<b><u>Beschreibung</u></b>
FR-015	Ausgeben der Werkstück-Eigenschaften	Anlage	Wenn ein Werkstück das Ende von Band 2 erreicht hat, werden die ID, der Typ und die Höhenmesswerte von Band 1 und Band 2 auf der Konsole ausgegeben.
FR-016	Ampelanlage	Anzeige	Die Werkstück-Sortieranlage besitzt an beiden Bändern jeweils eine Ampelanlage, mit der sich der Betriebszustand des jeweiligen Bandes und der gesamten Anlage abbilden lässt.
FR-017	Grüne Signalleuchte	Anzeige	Die grüne Signalleuchte der jeweiligen Ampelanlage signalisiert den fehlerfreien Betrieb.
FR-018.1	Gelbe Signalleuchte	Anzeige	Die gelbe Signalleuchte von Band 1 signalisiert einen Hinweis an das Personal das Werkstück von Hand mit der Bohrung nach oben zu drehen.
FR-018.2	Gelbe Signalleuchte	Anzeige	Die gelbe Signalleuchte von Band 2 signalisiert einen Hinweis an das Personal das Werkstück vom Ende des Bandes zu entfernen.
FR-019.1	Rote Signalleuchte	Anzeige	Die Anlage besitzt eine rote Signalleuchte um zu signalisieren, dass ein Fehler aufgetreten ist: Rutsche voll; zu lange Laufzeit; zu kurze Laufzeit.
FR-019.2	Rote Signalleuchte	Anzeige	Ein nicht quittierter Fehler lässt die rote Signalleuchte schnell blinken (1Hz).
FR-019.3	Rote Signalleuchte	Anzeige	Die Quittierung eines Fehlers ändert das Blinken der roten Signalleuchte in ein Dauerlicht.
FR-019.4	Rote Signalleuchte	Anzeige	Ein Fehler, der verschwunden ist oder sich von selbst gelöst hat, lässt die rote Signalleuchte langsam blinken (0.5Hz).
FR-019.5	Rote Signalleuchte	Anzeige	Solange keine Fehler aufgetreten sind, ist die rote Signalleuchte erloschen.
FR-020	Ruhezustand	Anlage	Befinden sich keine Werkstücke auf den Bändern, sollen diese angehalten werden.
FR-021	Verschwinden von Werkstücken	Fehler	Bei zu langen Laufzeiten zwischen den Lichtschranken liegt ein Verschwinden des Werkstücks vor. Es wird eine Fehlermeldung ausgegeben und die Anlage stoppt.
FR-022	Hinzufügen von Werkstücken mitten auf dem Band	Fehler	Bei zu kurzen Laufzeiten zwischen den Lichtschranken wurde ein Werkstück mitten auf das Band gelegt. Es wird eine Fehlermeldung ausgegeben und die Anlage stoppt.



<u>ID</u>	<u>Titel</u>	<u>Bereich</u>	<u>Beschreibung</u>
FR-023	Rutsche voll	Fehler	Bei zu vielen Werkstücken in der Rutsche für die Aussortierung fehlerhafter Werkstücke wird eine Fehlermeldung ausgegeben und die Anlage gestoppt.
FR-024.1	Ansteuerung der Weiche	Weiche	Die Weiche ist im geschlossenen Zustand stromlos und führt Strom, wenn sie geöffnet ist.
FR-024.2	Ansteuerung der Weiche	Gefahr	Eine dauerhaft geöffnete Weiche muss aufgrund von Überhitzung des Motors vermieden werden!
FR-025	Not-Aus der Anlage	Gefahr	Die Anlage darf erst nach einem Reset und einem erneuten Starten durch den Start-Taster wieder in Betrieb gehen und nicht bereits nach der Behebung des Fehlers!
FR-026	Bandlaufgeschwindigkeit bei Höhenmessung	Sensorik	Bei der Höhenmessung auf Band 1 und Band 2 werden die Werkstücke im langsamen Modus des Bandes befördert.
FR-027	Start-Taster	Taster	Die Anlage besitzt einen Start-Taster samt zugehöriger Leuchte zum Einschalten der Anlage.
FR-028	Stop-Taster	Taster	Die Anlage besitzt einen Stop-Taster samt zugehöriger Leuchte zum Ausschalten der Anlage.
FR-029.1	Reset-Taster	Taster	Die Anlage besitzt einen Reset-Taster samt zugehöriger Leuchte zur Quittierung von Fehlern im Betrieb der Anlage.
FR-029.2	Reset-Taster	Taster	Jeder Fehler muss zuerst quittiert werden. Erst dann kann der Betrieb durch das Betätigen des Ein-Tasters wieder aufgenommen werden.
FR-030.1	E-Stopp-Taster	Taster	Die Anlage besitzt einen E-Stopp-Taster samt zugehöriger Leuchte zur Schnellabschaltung und Stilllegung der gesamten Werkstück-Sortieranlage.
FR-030.2	E-Stopp-Taster	Taster	Der E-Stopp-Taster ist LOW-aktiv.

### 1.10.2 Non-functional Requirements

<b>ID</b>	<b>Titel</b>	<b>Bereich</b>	<b>Beschreibung</b>
NFR-001	Timer für das Wenden eines Werkstücks am Ende von Band 1	Sensorik	Nachdem erkannt wird, dass ein Werkstück mit der Bohrung nach unten auf Band 1 liegt, wird es in die Lichtschranke am Ende von Band 1 gefahren und es läuft ein Timer für 60 Sekunden. In dieser Zeit muss das Werkstück vom Personal gewendet werden, ansonsten wird eine Fehlermeldung ausgelöst.
NFR-002	Timer für das Entfernen eines Werkstücks in falscher Reihenfolge am Anfang von Band 2	Sensorik	Nachdem das korrekt sortierte Werkstück in falscher Reihenfolge auf Band 2 erkannt wurde, fährt es in die Lichtschranke am Anfang von Band 2 zurück und es läuft ein Timer für 60 Sekunden. In dieser Zeit muss das Werkstück vom Personal entfernt werden, ansonsten wird eine Fehlermeldung ausgelöst.
NFR-003	Timer für das Entfernen eines korrekt sortierten Werkstücks am Ende von Band 2	Sensorik	Nachdem das korrekt sortierte Werkstück das Ende der Lichtschranke von Band 2 erreicht hat, läuft ein Timer für 60 Sekunden. In dieser Zeit muss das Werkstück vom Personal entfernt werden, ansonsten wird eine Fehlermeldung ausgelöst.
NFR-004.1	Timer für die Beförderung eines Werkstücks durch die Werkstück-Sortieranlage	Sensorik	Ein korrektes Werkstück durchläuft die einzelnen Bänder der Werkstück-Sortieranlage jeweils innerhalb eines Intervalls von 4-5 Sekunden.
NFR-004.2	Timer für die Beförderung eines Werkstücks durch die Werkstück-Sortieranlage	Sensorik	Eine Laufzeitmessung durch die Lichtschranken von weniger als 4-5 Sekunden bedeutet, dass ein weiteres Werkstück unter Fremdeinwirkung auf das Band gelegt worden ist. Das Band wird angehalten und es wird eine Fehlermeldung ausgelöst.
NFR-004.3	Timer für die Beförderung eines Werkstücks durch die Werkstück-Sortieranlage	Sensorik	Eine Laufzeitmessung durch die Lichtschranken von mehr als 4-5 Sekunden bedeutet, dass ein Werkstück unter Fremdeinwirkung vom Band entfernt worden ist. Das Band wird angehalten und es wird eine Fehlermeldung ausgelöst.
NFR-005	Timer für das Hinzufügen neuer Werkstücke auf Band 1	Sensorik	Nachdem ein Werkstück in die Lichtschranke am Anfang von Band 1 gelegt worden ist und das Band läuft, startet ein Timer für 3 Sekunden. In dieser Zeit darf kein neues Werkstück in die Lichtschranke am Anfang von Band 1 gelegt werden, da sonst ein reibungsloser Betrieb nicht mehr garantiert ist. Ansonsten wird eine Fehlermeldung ausgelöst.
NFR-006.1	Toleranz für die Höhe eines Werkstücks	Sensorik	Die Toleranz für die Höhe eines zu flachen Werkstücks liegt bei 10-15 Millimetern.
NFR-006.2	Toleranz für die Höhe eines Werkstücks	Sensorik	Die Toleranz für die Höhe eines Werkstücks mit Bohrung nach oben liegt bei 20-30 Millimetern.
NFR-006.3	Toleranz für die Höhe eines Werkstücks	Sensorik	Die Toleranz für die Höhe eines Werkstücks mit Bohrung nach unten liegt bei 25-30 Millimetern.
NFR-007	Öffnungsdauer der Weiche	Sensorik	Für das Durchlassen korrekter Werkstücke wird die Weiche für 2 Sekunden lang geöffnet.

## 1.11 Use Cases

### 1.11.1 ID: UC1

**Titel:** Akzeptiertes Werkstück  
**Akteur:** Personal  
**Ziel:** Werkstück kommt am Ende von Band 2 an.  
**Auslöser:** Ein Werkstück wird in die Lichtschranke am Anfang von Band 1 gelegt. B[0]=0

**Vorbedingung:**

1. Laufband 1 befindet sich im Betriebszustand.
2. Die Lichtschranke am Anfang von Band 1 ist frei. B[0]=1

**Nachbedingung:**

- keine

**Erfolgsszenario:**

1. Dem Werkstück wird eine ID vergeben.
2. Die Höhe des Werkstückes wird durch den Höhenmesser ermittelt. B[1]=0
3. Die Höhe des Werkstücks ist im Toleranzbereich. B[1]=1
4. Die Weiche des ersten Bandes wird geöffnet und das Werkstück durchgelassen. B[2]=0 und B[5]=1
5. Die Weiche wird geschlossen. B[5]=0
6. Das Werkstück kommt auf Band 2, wenn dieses frei ist.
7. Der Typ des Werkstücks wird festgelegt. Das Werkstück mit Bohrung nach oben und mit Metalleinsatz /ohne Metalleinsatz wird im Wechsel akzeptiert. (Metall-Kunststoff oder Kunststoff-Metall).
8. Die Weiche wird geöffnet und das Werkstück durchgelassen. B[2]=0 und B[5]=1
9. Die Weiche wird geschlossen. B[5]=0
10. Das Werkstück erreicht die Lichtschranke am Ende von Band 2. B[7]=0
11. Laufband 2 bleibt stehen.
12. Das Werkstück wird vom Personal entfernt. B[7]=1

**Fehlerszenario:**

1. Das Werkstück liegt nicht im Toleranzbereich der Höhe und wird aussortiert.
2. Die Kommunikation zwischen beiden Laufbändern funktioniert nicht.
3. Die Reihenfolge der Werkstücke ist falsch.
4. Werkstücke werden mitten im Betrieb hinzugefügt oder weggenommen.

### 1.11.2 ID: UC2

**Titel:** Nicht akzeptiertes Werkstück (zu flach)  
**Akteur:** -  
**Ziel:** Werkstücke die zu flach sind, werden von Band 1 aussortiert.  
**Auslöser:** Höhenmesser ermittelt die Höhe des Werkstücks. B[2]

**Vorbedingung:**

1. Laufband 1 befindet sich im Betriebszustand.
2. Eingelegtes Werkstück ist zu flach.

**Nachbedingung:**

- Aussortiertes Werkstück liegt in der Rutsche.

**Erfolgsszenario:**

1. Die Höhe des Werkstückes wird durch den Höhenmesser ermittelt. B[1]=0
2. Das Werkstück ist zu flach. B[2]=1
3. Die Weiche bleibt im geschlossenen Zustand. B[5]=0
4. Das Werkstück wird aussortiert.

**Fehlerszenario:**

1. Rutsche ist voll.
2. Werkstück bleibt in der Lichtschranke hängen.

### 1.11.3 ID: UC3

**Titel:** Nicht akzeptiertes Werkstück auf Band 1 (Bohrung nach unten)  
**Akteur:** Personal  
**Ziel:** Werkstück am Ende von Band 1 wird mit der Bohrung nach oben umgedreht.  
**Auslöser:** Höhenmesser ermittelt die Höhe des Werkstücks. B[2]

**Vorbedingung:**

1. Laufband 1 befindet sich im Betriebszustand.
2. Es befindet sich ein Werkstück auf Band 1.

**Nachbedingung:**

- Ein Werkstück befindet sich am Laufbandende.

**Erfolgsszenario:**

1. Die Höhe des Werkstückes wird durch den Höhenmesser ermittelt. B[1]=0
2. Ein Werkstück mit Bohrung nach unten wird erkannt.
3. Die Weiche wird geöffnet und das Werkstück durchgelassen. B[2]=0 und B[5]=1
4. Das Werkstück wird am Ende von Band 1 von der Lichtschranke registriert.
5. Das Laufband bleibt stehen und die gelbe Signalleuchte blinkt. A[6]=1
6. Das Personal dreht das Werkstück per Hand mit der Bohrung nach oben um.

#### 1.11.4 ID: UC4

**Titel:** Rutsche voll  
**Akteur:** Personal  
**Ziel:** Rutsche wieder frei. B[6]=1  
**Auslöser:** Sensorik erkennt, dass die Rutsche voll ist. B[6]=0

**Vorbedingung:**

1. Laufband befindet sich im Betriebszustand.
2. Die Rutsche ist voll.

**Nachbedingung:**

1. Die Rutsche ist wieder frei für mindestens ein Werkstück.
2. Das Laufband befindet sich im Betriebszustand. B[6]=1 und A[5]=1

**Erfolgsszenario:**

1. Laufband bleibt stehen. Rote Signalleuchte blinkt schnell (1Hz) → anstehend unquittiert. A[7]=1
2. Das Personal drückt den Reset-Taster → LED Resettaste leuchtet nicht mehr. C[1]=0
3. Rote Signalleuchte leuchtet (Dauerlicht) → anstehend quittiert. A[7]=1
4. Das Personal leert die Rutsche. B[6]=1
5. Das Personal bestätigt die Leerung der Rutsche durch Drücken des Start-Tasters.
6. Anlage läuft wieder → Rote Signalleuchte erlischt, grüne Signalleuchte leuchtet. A[7]=0 und A[5]=1

**Fehlerszenario:**

1. Das Personal leert die Rutsche, aber quittiert den Fehler nicht.
2. Der Start-Taster wird nicht nach Leerung der Rutsche gedrückt.
3. Der Fehler wird quittiert und die Anlage gestartet, ohne dass die Rutsche geleert wurde.

#### 1.11.5 ID: UC5

**Titel:** Verschwinden von Werkstücken  
**Akteur:** Personal  
**Ziel:** Das Fehlen des erfassten Werkstückes wird durch die Anlage signalisiert.  
**Auslöser:** Das Programm meldet zu lange Laufzeiten zwischen Lichtschranken.

**Vorbedingung:**

1. Laufband befindet sich im Betriebszustand.
2. Ein Werkstück wird vom Band entfernt.

**Nachbedingung:**

- Laufband läuft wieder (nur die grüne Signalleuchte leuchtet). A[5]=1

**Erfolgsszenario:**

1. Laufband bleibt stehen. Rote Signalleuchte blinkt schnell (1 Hz) → anstehend unquittiert. A[7]=1
2. Das Personal drückt den Reset-Taster → LED Resettaste leuchtet nicht mehr. C[1]=0
3. Rote Signalleuchte leuchtet nicht mehr.
4. Das Personal betätigt den Start-Taster.
5. Anlage läuft wieder → Grüne Signalleuchte leuchtet. A[7]=0 und A[5]=1

#### 1.11.6 ID: UC6

**Titel:** Hinzufügen von Werkstücken mitten auf dem Band  
**Akteur:** Personal  
**Ziel:** Das nicht erfasste Werkstück wird entfernt.  
**Auslöser:** Das Programm meldet zu kurze Laufzeiten zwischen Lichtschranken.

**Vorbedingung:**

1. Laufband befindet sich im Betriebszustand.
2. Ein Werkstück wird mittendrin auf das Band gelegt.

**Nachbedingung:**

- Laufband läuft wieder (nur die grüne Signalleuchte leuchtet). A[5]=1

**Erfolgsszenario:**

1. Laufband bleibt stehen. Rote Lampe blinkt schnell (1 Hz) → anstehend unquittiert. A[7]=1
2. Das Personal drückt den Reset-Taster → LED Resettaste leuchtet nicht mehr. C[1]=0
3. Rote Lampe leuchtet (Dauerlicht) → anstehend quittiert. A[7]=1
4. Das Personal entfernt das Werkstück.
5. Das Personal bestätigt das Entfernen des Werkstücks durch Drücken des Start-Tasters.
6. Anlage läuft wieder → Rote Signalleuchte erlischt, grüne Signalleuchte leuchtet. A[7]=0 und A[5]=1

**Fehlerszenario:**

1. Personal entfernt das Werkstück nicht.
2. Personal entfernt das falsche Werkstück.

#### 1.11.7 ID: UC7

**Titel:** Zurücksetzen des Laufbands  
**Akteur:** Personal  
**Ziel:** Laufband in Ursprungszustand versetzen.  
**Auslöser:** Das Personal betätigt den Reset-Taster. C[6]=1

**Vorbedingung:**

- Laufband befindet sich im Betriebszustand.

**Nachbedingung:**

- Auf der Anlage befindet sich kein Werkstück.

**Erfolgsszenario:**

1. Das Laufband bleibt stehen.
2. Das Personal entnimmt alle Werkstücke von der Anlage.
3. Das Personal betätigt erneut die Reset-Taste. C[6]=0
4. Laufband ist Betriebsbereit → Grüne Signalleuchte leuchtet. A[5]=1

#### 1.11.8 ID: UC8

**Titel:** Starten der Anlage nach Schnellabschaltung  
**Akteur:** Personal  
**Ziel:** Die gesamte Anlage ist wieder Betriebsbereit.  
**Auslöser:** E-Stopp-Taster wird gedrückt. C[7]=0

**Vorbedingung:**

- Anlage ist angeschaltet.

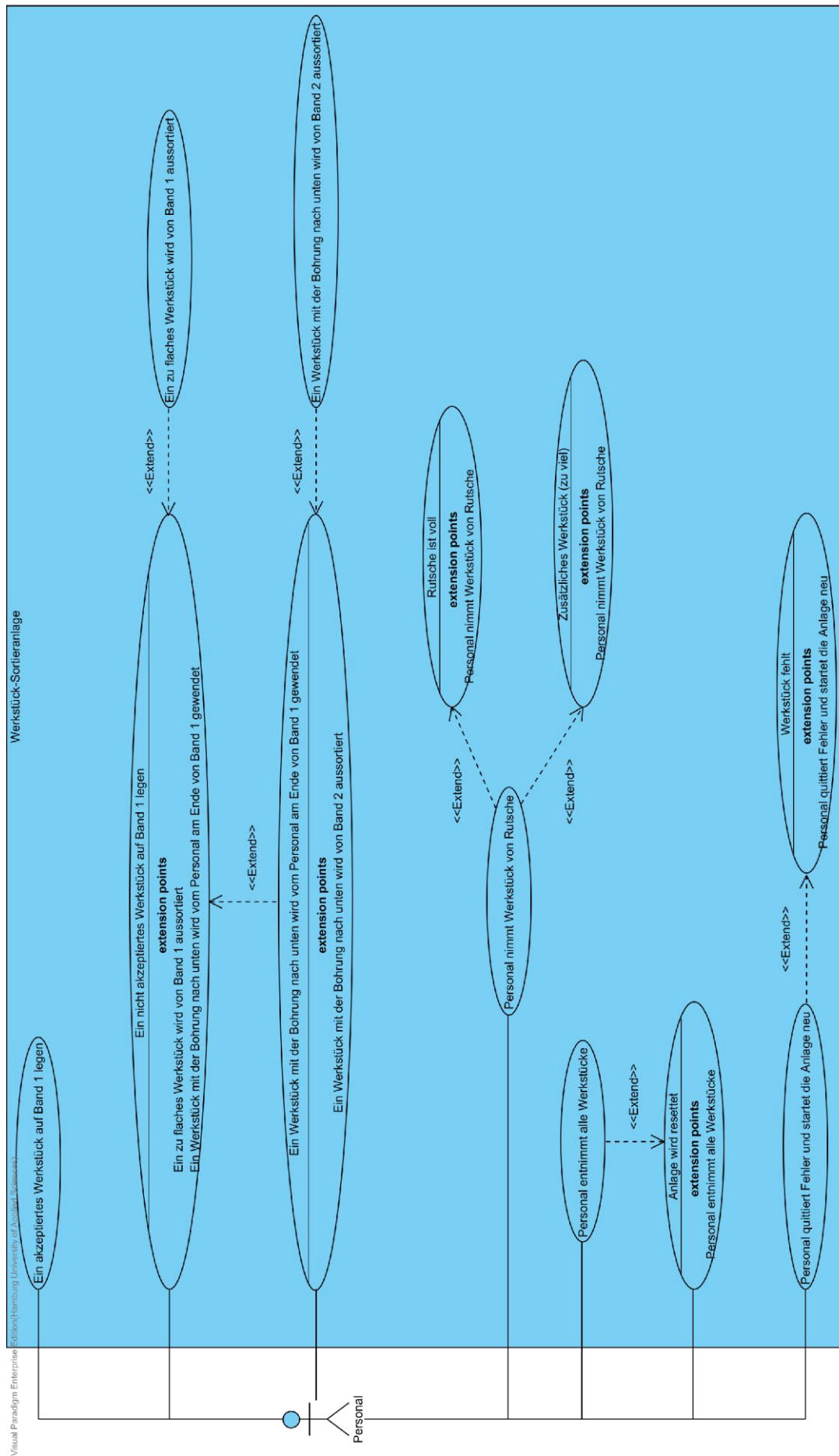
**Nachbedingung:**

- Laufband ist Betriebsbereit. A[5]=1

**Erfolgsszenario:**

1. Die ganze Anlage (Band 1 und Band 2) steht still.
2. Alle Ampeln sind aus.
3. Alle Weichen sind geschlossen.
4. Das Personal drückt den Start-Taster.
5. Die Anlage läuft wieder.

## 1.12 Use-Case-Diagramm





### 1.13 Systemanalyse

Was muss man über das technische System (aus Sicht der zu entwickelnden Software) wissen? Wie sieht die Struktur aus? Wie der Systemkontext? Welche Schnittstellen betrachten Sie?

## Design

Anmerkung: Die Implementierung MUSS mit Ihrem Design-Modell korrespondieren. Daher ist ein wohlüberlegtes Design wichtig.

### 1.14 System Architektur

Erstellung der System-Architektur. Geben Sie eine kurze Beschreibung Ihrer Architektur mit den dazugehörigen Komponenten und Schnittstellen.

Spezifikation der Architektur und Definition der System-Schnittstellen in einem UML Komponentendiagramm.

### 1.15 Datenmodell

Bestimmung des Datenmodells mit Hilfe von UML Klassendiagrammen unter Beachtung der Designprinzipien.

Kurze textuelle Beschreibung des Datenmodells und deren wichtigsten Klassen und Methoden.

### 1.16 Verhaltensmodell

Spezifikation der wichtigsten System-Szenarien anhand von Verhaltensdiagrammen.

Sie können für die Spezifikation der Prozess-Lenkung entweder Petri-Netze oder hierarchische Automaten nehmen.

## Implementierung

Anmerkung: Wichtige Implementierungsdetails sollen hier erklärt werden. Code-Beispiele (snippets) können hier aufgelistet werden, um der Erklärung zu dienen.

Anmerkung: Bitte KEINE ganze Programme hierhin kopieren!

### 1.17 Algorithmen

Wichtige Algorithmen, die Sie hier benutzt haben.

### 1.18 Patterns

Wichtige Patterns, die Sie implementiert haben.

### 1.19 Mapping Rules

Wichtige Mapping Rules, die Sie benutzt haben, z.B. um aus Ihrem Design entsprechenden Code zu erstellen.

## Testen

Machen Sie sich Gedanken über Unit-Test, Komponententest, Integrationstest, Systemtest, Regressionstest und Abnahmetest.

## 1.20 Unit Test/Komponenten Test

Test Szenario eines Laufbands.

## 1.21 Integration Test/System Test

Test Szenarien mit beiden Laufbändern

## 1.22 Regressionstest

Mögliche Regressionstests für serielle Schnittstelle:

- 1) korrektes Telegramm wird übertragen (Länge, Korrektheit)
- 2) fehlerhaftes Telegramm wird übertragen (ungültige Header-ID, ungültige Länge (EOF kommt zu früh oder zu spät, fehlerhafter Header, fehlerhafter Inhalt)
- 3) Synchronisierung

Weitere mögliche Regressionstests:

- 4) Hösensensor emulieren
- 5) Metallsensor emulieren

Welche Szenarien müssen immer wieder abgetestet werden? Automatisieren Sie Ihre Tests nach Möglichkeit

## 1.23 Abnahmetest

Zum Abnahmetest werden alle Requirements und Use Cases durchgegangen. Anschließend werden die Regressionstests und der Normalbetrieb der Anlage vorgeführt.

## 1.24 Testplan

Zeitpunkte für die jeweiligen Teststufen in Ihrer Projektplanung setzen. Dazu können Sie die Meilensteine zu Hilfe nehmen.

## 1.25 Testprotokolle und Auswertungen

Hier fügen Sie die Test Protokolle bei, auch wenn Fehler bereits beseitigt worden sind, ist es schön zu wissen, welche Fehler einst aufgetaucht sind. Eventuelle Anmerkung zur Fehlerbehandlung kann für weitere Entwicklungen hilfreich sein.

Das letzte Testprotokoll ist das Abnahmeprotokoll, das bei der abschließenden Vorführung erstellt wird. Es enthält eine Auflistung der erfolgreich vorgeführten Funktionen des Systems sowie eine Mängelliste mit Erklärungen der Ursachen der Fehlfunktionen und Vorschlägen zur Abhilfe

## Lessons Learned

Was lief gut, was lief schlecht in diesem Projekt (technisch und organisatorisch)?

Was haben Sie gelernt?

Weitere Anregungen und Erkenntnisse durch das Projekt.

## Glossar

Eindeutige Begriffserklärungen

## Abkürzungen

WS = Werkstück

Listen Sie alle Abkürzungen auf, die Sie in diesem Dokument benutzt haben.

## Anhänge

Auflistung aller Artefakte dieses Projekts.

1. Alle Modell-Dateien (Visual Paradigm, Petri-Netze etc.)
2. Source Code und Code Dokumentationen (z.B. Doxygen)
3. Test Protokolle
4. Meeting Protokolle
5. Projektplan
6. etc.

### 1.26 Coding Style: Google C++

#### 1.26.1 General rules

- Use 2 spaces per indentation level.
- The maximum number of characters per line is 80.
- Never use tabs.
- Vertical whitespaces separate functions and are not used inside functions: use comments to document logical blocks.
- Header filenames end in .hpp, implementation files end in cpp.
- Never declare more than one variable per line.
- and & bind to the type, e.g., `const std::string& arg`.
- Namespaces do not increase the indentation level.
- Access modifiers, e.g. `public`, are indented one space.
- Use the order `public`, `protected`, and then `private`.

- Use typename only when referring to dependent names.
- Keywords are always followed by a whitespace: if (...), template <...>, while (...), etc.
- Always use {} for bodies of control structures such as if or while, even for bodies consisting only of a single statement.
- Opening braces belong to the same line:

```
if (my_condition) {
    my_fun();
} else {
    my_other_fun();
}
```

- Use standard order for readability: C standard libraries, C++ standard libraries, other libraries, your headers:

```
#include <sys/types.h>
#include <vector>
#include "some/other/library.hpp"
#include "myclass.hpp"
```

- When declaring a function, the order of parameters is: outputs, then inputs. This follows the parameter order from the STL.
- Never use C-style casts.

### 1.26.2 Naming

- Class names, constants, and function names are all-lowercase with underscores.
- Types and variables should be nouns, while functions performing an action should be "command" verbs. Classes used to implement metaprogramming functions also should use verbs, e.g., **remove\_const**.
- Member variables use the prefix **m\_**.
- Thread-local variables use the prefix **t\_**.
- Static, non-const variables are declared in the anonymous namespace and use the prefix **s\_**.
- Template parameter names use CamelCase.

- Getter and setter use the name of the member without the `m_` prefix:

```
class some_fun {
public:
    // ...
    int value() const {
        return m_value;
    }
    void value(int new_value) {
        m_value = new_value;
    }
private:
    int m_value;
};
```

### 1.26.3 Headers

- Each .cpp file has an associated .hpp file. Exceptions to this rule are unit tests and main.cpp files.
- All header files should use #define guards to prevent multiple inclusion.
- Do not #include when a forward declaration suffices.
- Use inline for small functions (rule of thumb: 10 lines or less).

### 1.26.4 Breaking Statements

- Break constructor initializers after the comma, use four spaces for indentation, and place each initializer on its own line (unless you don't need to break at all):

```
my_class::my_class()
    : my_base_class(some_function()),
      m_greeting("Hello there! This is my_class!"),
      m_some_bool_flag(false) {
    // ok
}
other_class::other_class() : m_name("tommy"), m_buddy("michael") {
    // ok
}
```

- Break function arguments after the comma for both declaration and invocation:

```
intptr_t channel::compare(const abstract_channel* lhs,
                          const abstract_channel* rhs) {
    // ...
}
```

- Break before tenary operators and before binary operators:

```
if (today_is_a_sunny_day()
    && it_is_not_too_hot_to_go_swimming()) {
    // ...
}
```