ESEP – Praktikum – SS 2017

Requirements and Design Documentation

Sebastian Brückner, Stephan Jänecke

Version 0.2

Versionshistorie

Version	Datum	Autor(en)	Änderungen
0.1	30.03.17	Stephan Jänecke	Erster Entwurf
0.2	06.04.17	Sebastian Brückner	Testkonzept
0.2	06.04.17	Sebastian Brückner	Teamorga Entwurf
0.2	18.06.17	Stephan Jänecke	Systemkontextdiagramm

Inhaltsverzeichnis

1 Teamorganisation

Wir arbeiten Agil mit Scrum

1.1 Verantwortlichkeiten

Rolle	**Inhaber**
Scrum Master	Jonas Fuhrmann
Repository Beauftragter	René Herthel
Product Owner	Sebastian Brückner
Tex	Stephan Jänecke

1.2 Absprachen

1.3 Repository-Konzept

- Das Repository wird unterteilt in Dokumentation (Ordner doc) und Sourcen (Ordner src).
- Es wird git flow verwendet.
- Es werden keine Binaries eingecheckt

1.3.1 doc

Der Ordner Teilt sich in 4 Unterordner:

- **protocoll** Protokolle aller Sitzungen
- **rdd** LaTex Sourcen des RDD
- **templates** Templates für z.B. die Protokolle
- **uml** Dierekt im UML Ordner findet man die UML Diagramme der Architektur In den Unterordnern finden sich die UML Diagramme der Komponenten

2 Projektmanagement

In diesem Kapitel sollten organisatorische Punkte beschrieben und festgelegt werden.

2.1 Prozess

Legen Sie den Prozess fest, nach dem Sie das Projekt umsetzen wollen. Geben Sie ggf. grobe Schritte an, wie Planungsrunden, Sprints, oder ähnliches.

2.2 PSP/Zeitplan/Tracking

Projektstrukturplan, Ressourcenplan, Zeitplan, Abhängigkeiten von Arbeitspaketen, eventueller Zeitverzug, Visualisierung des Projektstandes, etc.

2.3 Qualitätssicherung

Überlegen sie, wie sie Qualität in ihrem Projekt sicher stellen wollen. Listen sie die Maßnahmen hier auf. Beachten sie, dass diese Maßnahmen für die unterschiedlichen Artefakte und Ebenen entsprechend unterschiedlich sein können.

3 Randbedingungen

3.1 Entwicklungsumgebung

Auflistung der Entwicklungsumgebung (Simulator, Hardware, Betriebssystem etc.)

3.2 Werkzeuge

Auflistung der im Projekt verwendeten Werkzeuge inkl. ihrer Versionen.

3.3 Sprachen

Auflistung der Programmiersprachen und Bibliotheken.

4 Requirements und Use Cases

4.1 Systemebene

Die Anforderungen aus der Aufgabenstellung sind nicht vollständig. Die Struktur der nachfolgenden Kapitel soll sie bei der Strukturierung der Analyse unterstützen. Dokumentieren Sie die Ergebnisse der Analysen entsprechend.

4.1.1 Stakeholder

Ermitteln sie die Stakeholder für das Projekt und listen sie diese hier auf.

4.1.2 Anforderungen

In der Aufgabenstellung sind Anforderungen an das System gestellt. Arbeiten sie diese hier auf und ergänzen sie diese entsprechend der Absprachen mit dem Betreuer. Achten sie auf die entsprechende Atribuierung. Berücksichtigen sie auch mögliche Fehlbedienungen und Fehlverhalten des Systems.

Requirements:

REQ – 01	Laufband starten

Die Software soll das Laufband starten sobald ein Werkstück auf den Anfang gelegt wird.

REQ – 02	Werkstücke sortieren

Die Software sortiert Werkstücke laut dem vorgegebenen Schema: Bohrung oben ohne Metalleinsatz, Bohrung oben ohne Metalleinsatz, Bohrung oben mit Metalleinsatz.

REQ – 03	Durchlaufzeiten messen

Die Software misst die Durchlaufzeiten der Werkstücke, damit das Hinzufügen und verloren gehen von Werkstücken registriert wird.

REQ – 04	Weichen steuern

Die Software kann die Weichen öffnen und so die Werkstücke sortieren.

REQ – 05	Bänder stoppen

Die Software stoppt die Laufbänder, wenn siche keine Werkstücke darauf befinden.

REQ – 06	Fehler signalisieren

Die Software gibt bei Fehlern Fehlermeldungen auf der Konsole aus und signalisiert einen Fehler

zusätzlich durch die Rote Ampel.

REQ – 07	Werkstücke erkennen	Die Software erkennt verschiedene Typen von Werkstücken und gibt die Typkennung auf der Konsole aus sobald sie bekannt ist.

REQ – 08	Kommunikation	Die Software ermöglicht Kommunikation zwischen zwei Laufbändern.

REQ – 09	Volle Rutschen	Die Software erkennt wenn die Rutschen zum Aussortieren voll sind und Sortiert die Werkstücke dann auf dem anderen Band.

REQ – 10	Mehrerer Werkstücke	Die Software kann mehrere Werkstücke auf dem ersten Laufband bearbeiten und sorgt dafür, dass sich nur ein einziges Werkstück zur Zeit auf dem zweiten Laufband befindet.

REQ – 11	Sortierte Werkstücke	Die Software gibt für Werkstücke die das Ende des zweiten Laufbandes erreichen, die ID, den Typ, Höhenmesswert 1 und Höhenmesswert 2 auf der Konsole aus.

REQ – 12	Bedientaster	Die Software kann mit den Bedientastern Ein/Aus geschaltet werden und verfügt über eine E-Stop Funktion, die die komplette Anlage zum Stillstand bringt.

REQ – 13	Geschwindigkeit anpassen	Die Software passt die Geschwindigkeit der Laufbänder so an das sie langsam durch die Höhenmessung fahren.

4.1.3 Systemkontext

Die Sicht, aus der die Use Cases verfasst wurden, geht aus Abbildung 4.1 hervor.

4.1.4 Use Cases

Dokumentieren sie hier, welche Use Cases sie auf der Systemebene implementieren müssen. Die Test Cases sollen später zu den Use Cases konsistent sein.

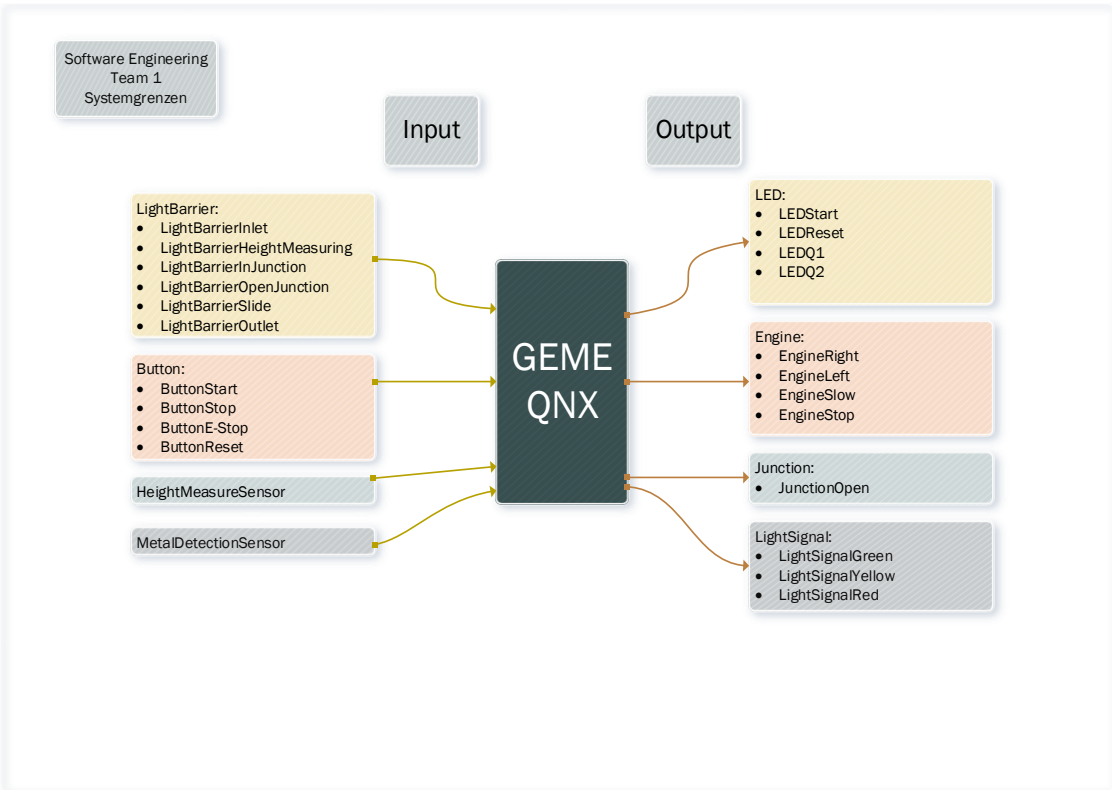

Abbildung 4.1: Systemkontextdiagramm

ID	UC-01
Name	Werkstück sortieren
Autoren	Dennis Kochinky
Priorität	Hoch
Verantwortliche	ESEP-Team 1.2
Akteure	Band, Lichtschranken, Weichen, Steuerungssoftware, User
Kurzbeschreibung	Ein Werkstück wird auf das Band, vermessen und einsortiert.
Auslösendes Ereignis	Werkstück wird auf das erste Laufband gelegt.
Vorbedingung	Anfang des ersten Bandes ist frei, Anlage ist Betriebsbereit.
Nachbedingung	Werkstück hat Ende des zweiten Bandes erreicht.
Ergebnis	Werkstück einsortiert
Hauptszenario	1. Werkstück wird auf Band 1 gelegt. 2. Die Software erkennt Unterbrechung der Lichtschranke und startet das Band. 3. Werkstück erreicht im vorgegeben Zeitfenster die Höhenmessung und wird analysiert. 4. Weiche wird geöffnet. 5. Werkstück wird auf Band 2 übergeben. 6. Erfassung, Höhenmessung des Werkstückes und öffnen der Weiche auf Band 2. 7. Werkstück hat das Ende von Band 2 erreicht.
Alternativszenario 1	4a) Das Werkstück ist zu flach, die Weiche bleibt geschlossen und das Werkstück wird auf die Rutsche geleitet.
Alternativszenario 2	6a) Werkstück wird für die Sortierung nicht gebraucht, die Weiche auf Band 2 bleibt geschlossen und das Werkstück wird auf Rutsche 2 aussortiert.
Fehlerszenario	3a) Werkstück ist zu schnell (ein weiteres Werkstück wurde zwischendurch aufs Band gelegt) oder Werkstück ist zu langsam (ein Werkstück wurde entfernt), das Band wird gestoppt.

ID	UC-02
Name	Rutsche voll.
Autoren	Dennis Kochinky
Priorität	Mittel
Verantwortliche	ESEP-Team 1.2
Akteure	Band, Lichtschranken, Weichen, Steuerungssoftware, User
Kurzbeschreibung	Die Software erkennt, dass eine der Rutschen voll ist und setzt das Aussortieren auf dem anderen Band mit freier Rutsche fort
Auslösendes Ereignis	Die Lichtschranke an der Rutsche ist dauerhaft unterbrochen.
Vorbedingung	Werkstücke sind auf dem Band.
Nachbedingung	Sortierung wird auf dem Band mit freier Rutsche fortgesetzt.
Ergebnis	Dem User wird die volle Rutsche signalisiert und Sortierung läuft weiter.
Hauptszenario	1. Ein Werkstück wird auf Band 1 gelegt. 2. Die Software erkennt, dass die Rutsche voll ist und signalisiert dies durch die gelbe Ampel. 3. Die Weiche wird geöffnet. 4. Werkstück erreicht Ende von Band 1 und wird auf Band 2 übergeben. 5. Sortierung wie in UC-01.
Alternativszenario	2.1a Beide Bänder melden Rutsche voll. 2.1b Band wird gestoppt und Fehler signalisiert.

ID	UC-03
Name	Kalibrierung
Autoren	Dennis Kochinky
Priorität	Hoch
Verantwortliche	ESEP-Team 1.2
Akteure	Band, Lichtschranken, Weichen, Steuerungssoftware, User
Kurzbeschreibung	System wird nach dem einschalten kalibriert, dazu wird ein Werkstück auf das Band gelegt.
Auslösendes Ereignis	System wird eingeschaltet und ein Werkstück wird vom User auf das erste Band gelegt.
Vorbedingung	Das System ist im Startzustand.
Nachbedingung	Alle Sensoren und Aktore funktionieren.
Ergebnis	Werkstück ist am Ende von Band 2 angekommen.
Hauptszenario	1. Das System wird eingeschaltet. 2. Ein Werkstück wird auf Band 1 gelegt. 3. Die Software erkennt Unterbrechung der Lichtschranke und startet das Band (Timer). 4. Werkstück erreicht die Höhenmessung, Bandgeschwindigkeit wird verlangsamt. 5. Die Weiche wird geöffnet. 6. Werkstück erreicht Ende von Band 1 und wird auf Band 2 übergeben. 7. Daten werden auf der Konsole ausgegeben. 8. Weiter mit Band 2.
Fehlerszenario	Band oder Weiche kann nicht angesteuert werden, Werkstück landet auf Rutsche.

ID	UC-04
Name	E-Stopp
Autoren	Dennis Kochinky
Priorität	Mittel
Verantwortliche	ESEP-Team 1.2
Akteure	E-Stopp-Taster, Band, Weichen, User
Kurzbeschreibung	Betätigung des E-Stopp-Tasters sorgt für Abschaltung der Bänder.
Auslösendes Ereignis	E-Stopp-Taster wird vom User gedrückt
Vorbedingung	Das System in Betrieb.
Nachbedingung	Bänder stehen still und Weichen sind geschlossen.
Ergebnis	Komplette Anlage steht still.
Hauptszenario	1. E-Stopp-Taster wird vom User gedrückt. 2. Die Bänder werden gestoppt. 3. Die Weichen werden geschlossen. 4. Die Ampel wird ausgeschaltet. 5. Warten auf Reset.

4.2 Systemanalyse

Ihr technisches System hat aus Sicht der Software bestimmte Eigenschaften. Was muss man für die Entwicklung der Software in Struktur, Schnittstellen, Verhalten und an Besonderheiten wissen? Wählen sie eine Kapitelstruktur, die am besten zur Dokumentation ihrer Ergebnisse geeignet ist.

4.3 Softwareebene

Sie sollen Software für die Steuerung des technischen Systems erstellen. Aus den Anforderungen auf der Systemebene und der Systemanalyse ergeben sich Anforderungen für Ihre Software. Insbesondere wird sich die Software der beiden Anlagenteile in einigen Punkten unterscheiden. Dokumentieren sie hier die Anforderungen, die sich speziell für die Software ergeben haben.

4.3.1 Systemkontex

Wie sieht der Kontext Ihrer Software aus? Wie erfolgt die Kommunikation mit Nachbarsystemen? Liste der ein- und ausgehenden Signale/Nachrichten.

4.3.2 Anforderungen

Welche wesentlichen Anforderungen ergeben sich aus den Systemanforderungen für ihre Software? Achten sie auf die entsprechende Atribuierung. Berücksichtigen sie auch mögliche Fehlbedienungen und Fehlverhalten des Systems.

5 Design

Anmerkung: Die Implementierung MUSS zu Ihrem Design-Modell konsistent sein. Strukturen, Verhalten und Bezeichner im Code müssen mit dem Modell übereinstimmen. Daher ist ein wohlüberlegtes Design wichtig.

5.1 System Architektur

Erstellung sie eine Architektur für Ihre Software. Geben sie eine kurze Beschreibung Ihrer Architektur mit den dazugehörenden Komponenten und Schnittstellen an. Dokumentieren sie hier wichtige technische Entscheidungen. Welche Pattern werden gegebenenfalls verwendet? Wie erfolgt die interne Kommunikation?

5.2 Datenmodellierung

Bestimmen sie das Datenmodell und dokumentieren sie es hier mit Hilfe von UML Klassendiagrammen unter Beachtung der Designprinzipien. Die Modelle können mit Hilfe eines UML-Tools erstellt werden. Hier ist dann ein Übersichtsbild einzufügen. Geben sie eine kurze textuelle Beschreibung des Datenmodells und deren wichtigsten Klassen und Methoden an.

5.3 Verhaltensmodellierung

Ihre Software muss zur Bearbeitung der Aufgaben ein Verhalten aufweisen. Überlegen sie sich dieses Verhalten auf Basis der Anforderungen und modellieren sie das Verhalten unter Verwendung von Verhaltensdiagrammen. Sie können für die Spezifikation der Prozess-Lenkung entweder Petri-Netze oder hierarchische Automaten verwenden. Die Modelle können mit Hilfe eines UML-Tools erstellt werden. Hier sind dann kommentierte Übersichtsbilder einzufügen.

6 Implementierung

Anmerkung: Nur 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!

7 Testen

7.1 Test Konzept

Das Testen soll der kleinsten zur Größten Einheit ablaufen. Alle Klassen werden gegen Basis Klassen implementiert (Design for Testability).

7.1.1 Unit Test

Nach dem fertig stellen einer Klasse sollte dieses mithilfe eines Unit Tests geprüft werden (simple Funktionen wie z.B setter und getter müssen nicht getestet werden). Nach dem ändern dieser Klasse ist der Unit Test nochmals durchzuführen (Regressionstest). Zur Verifikation des Unit Test sind über einen Whitebox-Test folgende Metriken zu beachten:

- **Code Coverage:** Diese muss bei 100% liegen, daraus folgt auch eine 100%Branch coverage.
- **Condition Coverage:** Komplexe Conditions sollten mit einer Kompletten Wertetabelle getestet werden.

7.1.2 Komponenten Test

Nach dem Fertigstellen einer Komponente soll diese, mithilfe des nach außen Sichtbaren Interfaces zuerst mit Stubs und Mocks (ohne Hardware) auf ihre Funktionalität geprüft werden. Ist dieser Test erfolgreich, so wird der Test erneut auf der Hardware durchgeführt ohne Stub und Mock Objekte in der Komponente.

Nach ändern der Komponente sind die Tests erneut durchzuführen.

7.1.3 Integrationstest

Fast alle Komponenten kommuniezieren zentral über den Main Controller. Je nach dem ob eine Komponente Input für den Controller oder vom Controller gestuert wird(Output), sind folgende Test Durchzuführen. Bei Komponenten mit in und Output mit In- und Output ist beides zu beachten. Zuerst werden diese Test in einer Testumgebung (ohne Hardware) durchgeführt, dabb auf dem echten Anlagen.

Input

Erhält der Controller die spezifizierten Signale von der Komponente? Liefert die Komponente die richtigen Signale zum Hardware Input?

Output

Verhält sich die Komponente entsprechend zum Eingangssignal?

Test der MainControll

Die MainControll wird mit Moch/Stub/Fake und Spy Objekten umgeben, die das Interface der Komponenten implementieren. Zuerst ist sicherzustellen, das alle Basisfunktionen einzeln funktionieren. Dazu gehören:

- Transport der Pucks
- Höhenmessung
- Sortieren der Pucks
- Rampe Voll
- Übergabe an die hintere Anlage (nur bei Anlage 1);
- E-Stopp
- Fehlerbehandlung
  - Ampelanlage
  - Knöpfe

7.1.4 System Test

Jedem Requirement wird mindestens ein Test zugewiesen. Dieser Test wird auf der Gesamten Anlage durchgeführt. Die Zuordnung von Requirement auf Test erfolgt mithilfe einer Trace Matrix. Hier wird eingetragen, ob der Test zum Requirement bestanden wurde oder nicht.

7.2 Testplan

Unit Tests

Der Test zu einer Unit soll Parallel zur Unit entwickelt werden. Sobald beide fertig sind, ist der Test durchzuführen.

Komponenten Test

Der Test zu einer Komponente soll vor der Komponente fertig sein, damit mit die Komponente Test-Driven entwickelt werden kann. Der Test ist nach jedem Entwicklungsschritt der Komponente durchzuführen. An ihm soll sich messen lassen, wie weit die Komponente fortgeschritten ist.

Integrationstest

Mit der MainControl zu entwickeln.

Systemtest

Nach fertigstellung aller Komponenten.

7.3 Abnahmetest

Leiten sie die Abnahmebedingungen aus den Kunden-Anforderungen her. [Ist das nicht Systemtest?] Dokumentieren sie hier, welche Schritte für die Abnahme erforderlich sind und welches Ergebnis jeweils erwartet wird (Test Cases).

7.4 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.

8 Lessons Learned

Führen sie ein Teammeeting durch in dem gesammelt wird, was gut gelaufen war, was schlecht gelaufen war und was man im nächsten Projekt (z.B. im PO) besser machen will. Listen sie für die Aspekte jeweils mindestens drei Punkte auf. Weitere Erfahrungen und Erkenntnisse können hier ebenso kommentiert werden, auch Anregungen für die Weiterentwicklung des Praktikums.

9 Anhang

9.1 Glossar

Eindeutige Begriffserklärungen

9.2 Abkürzungen

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

9.3 Signale

Einen Überblick über die Signale und die beteiligten Module bietet Tabelle 9.1.

9.4 Pulse Message Module IDs

In Tabelle 9.2 wird jedem Modul ein eindeutiger Identifier zugewiesen. Diese werden in den Pulse Messages genutzt, um die Nachrichten den Modulen zuordnen zu können.

Signal	Sender	Empfänger
LightMessage	ILightSystem	LightSystemController

Tabelle 9.1: Signale zwischen den Modulen

ID	Modul
80	LightSystem

Tabelle 9.2: Pulse Message Identifier für die Module