Werkstück Sortieranlage

Requirements and Design Documentation

(RDD)

Version 0.21

ESEP – Praktikum – SS 2016

Haidarbaigi, Diana, 2160484, Diana.Haidarbaigi@haw-hamburg.de

Kessener, Daniel, 2159858, Daniel.Kessener@haw-hamburg.de

Schuler, Malte, 2227928, Malte.Schuler@haw-hamburg.de

Änderungshistorie:

Version	Autor	Datum	Anmerkungen/Änderungen
0.1	*Thomas Lehmann*	*2014-09-17*	*Überarbeitung des Templates*
0.2	*Thomas Lehmann*	*2015-09-20*	*Einarbeitung von Arbeitsaufträgen und Erwartungen in die Kommentare*
0.3	*Thomas Lehmann*	*2015-09-28*	*Entfernung überflüssiger Kapitel, Detailierung einiger Beschreibungen.*
0.4	*Thomas Lehmann*	*2016-03-11*	*Anpassung an ESEP*
0.4	Malte Schuler	2016-04-05	Einarbeitung persönlicher Details sowie Randbedingungen
0.5	Malte Schuler	2016-04-05	Teamorganisation Teil Update
0.51	Malte Schuler	2016-04-05	Formatierung und Automatisches Inhaltsverzeichnis

0.6	Diana Haidarbaigi	2016-04-05	Teamorganisation vorerst abgeschlosse Stakeholder und Anforderungsanalyse erarbeitet
0.61	Diana Haidarbaigi	2016-04-06	Anforderungen gemäß der Absprachen überarbeitet

Inhalt

Teamorganisation

Nächster Termin	Uhrzeit	Milestone
11.04.16	16:00	2

Nach jedem Praktikum Treffen

Überlegen sie, welche Regeln sie für die Zusammenarbeit aufstellen wollen und welche Rollen sie im Team verteilen wollen. Dokumentieren sie diese hier zusammen mit weiteren Anmerkungen der Teamorganisation. Listen oder Tabellen sind zum Beispiel ein kompakte und übersichtliche Darstellungsformen für diesen Bereich.

1. Verantwortlichkeiten

- Software Design: Alle
- Implementierung: Daniel Kessener
- Tester: Malte Schuler
- Dokumentation: Diana Haidarbaigi

Bennen sie Verantwortliche innerhalb des Projekts (Projektleiter, Tester, Implementierer, etc.). Auch hier ist eine Listen- oder Tabellendarstellung angebracht.

2. Absprachen

Versionsverwaltung

Das Team benutzt das auf GitHub erstellte Repository zur Versionsverwaltung. Durch ein oftmaliges ausführen von Commits in jedem Schritt kann sicher gestellt werden, das bei Problemen auf die zuletzt lauffähige Version zurückgegriffen werden kann.

GitHub-Repository: https://github.com/davekessener/TI4_SE2.git

Für Absprachen und Probleme bezüglich des Projektes und des Ablaufes wird, wenn möglich, das persönliche Gespräch ersucht.

Kommunikation

Sollte dies aufgrund von terminlichen oder zeitlichen Aspekten nicht möglich sein, so soll über „Slack“ oder per Mail Kontakt aufgenommen werden. Für die Organisation ist es wichtig, dass jedes Teammitglied in kurzer Zeit über Slack erreichbar ist.

Dies kann durch das Installieren von Slack auf oft benutzten Medien wie Handy oder Computer sichergestellt werden, da diese für sehr viele Plattformen verfügbar ist.

Dadurch wird die Respond-Latenz möglichst gering gehalten und wird vermutlich maximal einen Tag erreichen.

Immer nach dem Praktikum und ansonsten nach Absprache.

Arbeitsrespondzeiten

Probleme/Bugs werden gemeinsam priorisiert und es wird ein Abarbeitungsplan erstellt.

3. Repository-Konzept

Für die Dokumentation des Projektes wird ein separates Projekttagebuch geführt, dieses liegt auf Papierbasis vor.

Zusätzlich davon wird das RDD per Git auf dem Laufenden gehalten. Es findet sich im Unterordner „doc“, dort sollen sich nach Möglichkeit auch weitere benötigte Dokumente wie z.B. die Projekt-Spezifikationen sammeln.

Folgend ist der Source Code Ordner „src“, dieser teilt sich selbst in zwei Unterordner auf. Im Ordner „src“ sind jedoch unteranderem noch der reguläre Source Code mit dem Projekt spezifischen Entwicklungen vorhanden.

Librarys – in diesem Falle „Hilfsfunktionen“ – werden im Unterordner „lib“ organisiert.

Unit Tests und andere Testreihen werden im Ordner test eingeordnet.

Im Ursprungsordner findet sich noch der Ordner „makefiles“, welcher für den Bau benötigte Anleitungen enthält.
Das Projektfile von Momentics, das Readme sowie shellscripte für Unit Tests werden ebenso im Ursprungsordner gehalten.

Die Commit-Nachrichten sollen den Sinn des Commits umfassen. Er soll auf Englisch gehalten werden.

Überlegen sie sich, wie sie das Repository und die Ordner organisieren wollen. Welche Regeln wollen sie beim Umgang mit Branches, Auslieferungen, Nachrichten an den Commits usw. im Team einhalten?. Listen sie diese Absprachen hier auf.

2. Projektplan

In diesem Kapitel sollten organisatorische Punkte beschrieben und festgelegt werden.

2.1. PSP/Zeitplan/Tracking

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

2.2. Qualitätssicherung

Überlegen sie, wie sie Qualität in ihrem Projekt sicherstellen 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

Simulator	
Hardware	GENE
Betriebssystem (Emb-HW)	QNX Neutrino RTOS
Betriebssystem (Comp-HW)	Microsoft Windows 7

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

3.2. Werkzeuge

Entwicklungsumgebung für QNX Neutrino RTOS:

QNX® Momentics® Integrated Development Environment
Version: 5.0.1
Build id: v201402230336

3.3. Sprachen

Programmiersprachen:

- C++ Version: C++03

Bibliotheken:

- HWaccess.h Version: Unknown Ursprung: Prof. Dr. Stephan Pareigis
- HAWThread Version: Unknown Ursprung: Prof. Dr. Stephan Pareigis
- Lock Version: 0.1 Ursprung: Simon Brummer
- concurrent Version: Unknown Ursprung: Daniel Kessener
- mpl Version: Unknown Ursprung: Daniel Kessener

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

Stakeholder	Interessen
Professoren	• Möglichst wenig Verlust an Teilnehmern • Vermittlung von Wissen durch praktische Anwendung
Studenten	• PVL Schein • neues Wissen • praktische Erfahrung • sinnvolles Zeitmanagement
Entwickler	• Präzise Anforderungen • Sorgfältige Dokumentation des Projekts • Sorgfältige Dokumentation der Testergebnisse
Tester	• „guter“ Stil von Entwicklern • Erfolgreiches Ausführen der Abnahmetests • ausführliche Requirements zur Testerstellung • Dokumentation der Testergebnisse
HAW	• Sorgfältiger Umgang mit den Laufbändern

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

4.1.2. Anforderungen

ID	Titel	Beschreibung
PE_01	Puk-Erkennung flach	Der Höhenmesser und Metall-Sensor sollen einen flachen Puk ohne Metalleinsatz erkennen.
PE_02	Puk-Erkennung Metalleinsatz Bohrung unten	Der Höhenmesser und Metall-Sensor sollen einen Puk mit Metalleinsatz und einer Bohrung die unten liegt erkennen.
PE_03	Puk-Erkennung Metalleinsatz Bohrung oben	Der Höhenmesser und Metall-Sensor sollen einen Puk mit Metalleinsatz und einer Bohrung die oben liegt erkennen.
PE_04	Puk-Erkennung OHNE Metalleinsatz Bohrung unten	Der Höhenmesser und Metall-Sensor sollen einen Puk OHNE Metalleinsatz und einer Bohrung die unten liegt erkennen.

ID	Titel	Beschreibung
PE_05	Puk-Erkennung OHNE Metalleinsatz Bohrung oben	Der Höhenmesser und Metall-Sensor sollen einen Puk OHNE Metalleinsatz und einer Bohrung die oben liegt erkennen.
PE_06	Erkennung eines nicht definierten Objektes	Wenn der Höhenmesser und der Metallsensor ein Objekt erkennen was nicht den aktzeptierten Puks entspricht, wird in einen Fehlerzustand gewechselt.
SORT	Sortierung nach Reihenfolge	Die Werkstücke sollen in der Reihenfolge flach, Bohrung oben ohne Metall, Bohrung oben mit Metall auf Band 2 ankommen
SORT_01	Aussortierung Bohrung unten	Puks, die die Bohrung unten haben, werden auf Band1 mit Hilfe der Höhenmessung erkannt und aussortiert
SORT_02	Aussortierung falsche Reiehnfolge -nicht metallisch Band1	Puks, die nicht der gewünschten Reihung entsprechen und nicht metallisch sind, werden auf Band 1 erkannt und aussortiert
SORT_03	Aussortierung falsche Reiehnfolge -metallisch Band 2	Puks, die nicht der gewünschten Reihung entsprechen und metallisch sind, werden auf Band 2 erkannt und aussortiert
SORT_04	Aussortierung falsche Reiehnfolge -Band 2	Puks die auf Band 2 nicht der gewünschten Reihenfolge entsprechen werden aussortiert und die gelbe Leuchte blinkt
SORT_05	Aussortierung Bohrung unten -Band 2	Puks die auf Band 2 mit Bohrung unten erkannt werden, werden aussortiert
LB_00	Laufband Tempo	Überall, außer bei der Höhenmessung ist das Tempo des Laufbandes normal.
LB_01	Zuführung von Puks	Die Zuführung der Puks erfolgt durch einlegen am Anfang von Band 1, bei der die Lichtschranke lb_entry unterbrochen werden muss. Der Abstand beträgt eine Puk-Länge (4cm) kurz vor der Höhenmessung
LB_02	Puk Identifizierung	Beim Erkennen des Puks am Anfang von Band 1 wird ihm eine ID zugewiesen, außerdem werden die Höhenmesswerte von Band 1 und Band 2 zu dem Puk gespeichert. Der Typ des Puks wird ebenfalls gespeichert und sollte, sofern die Bohrung nicht unten liegt, nach Durchlauf von Band 1 bekannt sein.
LB_03	Pukmenge Band1	Auf Band 1 dürfen sich mehrere Puks gleichzeitig hintereinander befinden.

ID	Titel	Beschreibung
LB_04	Pukmenge Band2	Es darf sich nur ein Puk zur Zeit auf dem Laufband befinden
LB_05	Übergang Band 1 zu Band 2	Es wird immer ein Puk einzeln auf das Band 2 transferiert und das auch nur, wenn Band 2 leer ist.
LB_06	Band 1 zu Band 2 Übergangsfehler	Beim Transferieren des Puks von Band 1 zu Band 2 kann es dazu kommen, dass sich der Puk überschlägt und damit die Bohrung andersherum auf dem Band liegt. Dieser Fehlerfall muss entsprechend behandelt werden
LB_07	Höhenmessung	Bei der Höhenmessung werden die Puks auf beiden Bändern langsam durchgeführt.
LB_08	Keine Puks	Beide Bänder sollen jeweils stoppen, wenn sich kein Puk auf dem Band befindet.
LB_09	Metall false-negative	Die Erkennung der Metalleinsätze ist fehleranfällig, deshalb muss dies durch eine Auswertung auf beiden Bändern kompensiert werden.
LB_10	volle Rutschen	Egal welche Rutsche voll ist, muss die Reihenfolge unter möglichst WENIG Ausschuss eingehalten werden.
L_11	Rutsche voll Band 1	Ist die Rutsche von Band 1 voll, so findet die Aussortierung auf Band 2 statt.
L_12	Rutsche voll Band 2	Ist die Rutsche von Band 2 voll, so findet die Aussortierung bereits auf Band 1 statt. Dieser Zustand muss dem Bediener signalisiert werden und die Pukmengenregel für Band2 muss stets eingehalten werden.
Fail_Puk_01	Puk weg	Verschwinden Puks (Erkennung durch zu lange Laufzeit zwischen Lichtschranken) muss das Band gestoppt werden und eine Fehlermeldung ausgegeben werden.
Fail_Puk_02	Puk dazugekommen	Wird ein Puk mitten auf dem Band hinzugefügt (Erkennung durch zu kurze Laufzeit durch Lichtschranken), das Band wird gestoppt und eine Fehlermeldung ausgegeben
Fail_slide_01	Beide Rutschen voll	Sind beide Rutschen voll stoppen die Bänder und es kommt zu einer Fehlermeldung

ID	Titel	Beschreibung
E_Stop_01	E_Stop wird gedrückt	Wird der E_Stop gedrückt steht die ganze Anlage, also Band 1 und Band 2 still. Also auch das Laufband wo der E_Stop nicht betätigt wurde. Wird der E_Stop herausgezogen bleiben beide Anlagen solange stehen, bis der Reset Knopf gedrückt wurde

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 Attribuierung.

Berücksichtigen sie auch mögliche Fehlbedienungen und Fehlverhalten des Systems.

4.1.3. Systemkontext

Use Cases werden aus einer bestimmten Sicht erstellt. Dokumentieren sie diese mittels Kontextdiagramm oder Use Case Diagramm. Die Use Cases und Test Cases müssen zu der hier verwendeten Nomenklatur konsistent sein.

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.

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.

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.

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!

6.1. Patterns

Wichtige Patterns, die sie implementiert haben.

6.2. Mapping Rules

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

7. Testen

Machen sie sich auf Basis ihrer Überlegungen zur Qualitätssicherung Gedanken darüber, wie sie die Erfüllung der Anforderungen möglichst automatisiert im Rahmen von Unit-Test, Komponententest, Integrationstest, Systemtest, Regressionstest und Abnahmetest überprüfen werden.

7.1. Abnahmetest

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

7.2. Testplan

Definieren sie Zeitpunkte für die jeweiligen Teststufen in ihrer Projektplanung. Dazu können sie die Meilensteine zu Hilfe nehmen.

7.3. 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.