Requirements / Design and Test Documentation (RDT)

Version 0.5

ESEP – Praktikum – Sommersemester 2023

Lorenz, Maik, 2542513, maik.lorenz@haw-hamburg.de

Schukow, Dominik, 2441109, dominik.schukow@haw-hamburg.de

Malik, Sulaiman, 2441151, sulaiman.malik@haw-hamburg.de

Änderungshistorie:

Version	Erstellt	Autor	Kommentar
0.1	2018-03-12	LMN	Initiale Version des Templates.
0.2	2020-03-15	DAI	Überarbeitung wegen Corona.
0.3	2022-02-24	LMN	Anpassungen für Sommersemester. Anforderungen an Requirements reduziert auf Ergänzungen.
0.4	2022-11-22 ff.	CHRS	Neustrukturierung des Templates, Schriftgrößen vereinheitlicht, Erweiterungen: Hinweise am Anfang des Dokuments, Unterkapitel Hardware und technische Gegebenheiten, Unterkapitel Analyse des Kundenwunsches , Unterkapitel Nachrichten und Signale, allg. Abnahmetest Text + Tabelle, Unterkapitel Abbildungsverzeichnis
0.5	2023-03-28		<weitere Versionen>

Inhaltsverzeichnis:

1 Teamorganisation

1.1 Verantwortlichkeiten

Verantwortlichkeit	Person/en
Projektmanager	Lorenz, Maik
Implementierung, Test	Schukow, Dominik
Requirements Analyst	Malik, Sulaiman

1.2 Absprachen

Dokumentation

- Source-Code auf GitHub
- Arbeitsversion RDT im SharePoint
- Abgabefertiges RDT im MS-Teams Raum für Gruppe 2.1

Kommunikation

- Feste Zeiten für Meetings im Labor: Donnerstag ab 12:00 Uhr bzw. nach dem Praktikum
- Freitag 13:00 Sprint-Planung / Standup
- Meetings je nach aktuellen Themen in MS-Teams
- Besprechungsprotokolle werden in Confluence dokumentiert

Aufgabenverteilung

- Über das Scrum-Board von JIRA

1.3 Repository-Konzept

Der Source-Code liegt auf GitHub im Projekt ESEP-2023SoSe-Team-2-1.

Wir arbeiten nach dem GitFlow Workflow. Im main-Branch dürfen nur funktionsfähige Versionen liegen. Im develop-Branch ist der Arbeitsstand für die nächste Version. Neue Features werden in eigenen feature-Branches implementiert und danach in den develop-Branch gemerged. Auslieferbereite Versionen werden mit Versionsnummern getaggt.

2 Projektmanagement

2.1 Prozess

Da wir nach dem Scrum-Modell arbeiten, werden zu bearbeitende Aufgaben immer in Sprints eingeplant. Sprints finden immer jeweils zwischen zwei Praktikumsterminen statt, dauern also in der Regel zwei Wochen.

Ein Review des gerade abgeschlossenen sowie die Planung eines neuen Sprints findet immer am Freitag 13:00 nach einem Praktikumstermin statt. In Wochen ohne Praktikum wird dieser Termin dazu genutzt, um den Stand der zu bearbeitenden Aufgaben des aktuellen Sprints zu besprechen.

Besprechungen werden immer schriftlich in Confluence dokumentiert. Sich daraus ergebende Absprachen werden ebenso dokumentiert und falls notwendig direkt in JIRA-Tasks eingeplant.

Wichtige Absprachen mit den Betreuern werden in diesem Dokument festgehalten.

2.2 Projektplan

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

Meilensteine

Zeitpunkt	Ziele
Praktikum 1	Organisation innerhalb des Teams definiert. Anforderungsanalyse und Systemkontextdiagramm erstellt. Projektplan und Projektstruktur erstellt. Momentics und Repository ist eingerichtet. Ein Programm kann auf die Anlage geladen werden und diese ansteuern
Praktikum 2	Vollständige A Anforderungsanalyse und Abmachungen sind dokumentiert. Die Aktorik der HAL ist implementiert. Beispiel zur Datenübertragung via QNET ist implementiert. Abnahmetests sind formuliert. Erstes Dokument der Software Architektur ist ausgearbeitet.
Praktikum 3	Überarbeitetes Dokument mit dem Entwurf der Software Architektur liegt vor. FSMs sind grob modelliert. Die Sensorik der HAL ist implementiert.

	Konzept der Übergabe der Daten von HAL zu FSM liegt vor. Präsentation der Architektur als Vortrag.
Praktikum 4	Das Dokument der Software Architektur ist final und kann implementiert werden.
Praktikum 5	FSMs sind ohne Fehlerbehandlung modelliert. Grundfunktionalität ohne Fehlerbehandlung implementiert (Werkstücke können sortiert werden)
Praktikum 6	FSMs sind vollständig modelliert. Die Anlage ist vollständig implementiert. Abgabe des finalen Requirement Design Dokument.
Praktikum 7	Alle nicht realisierten Funktionalitäten sind dokumentiert und begründet. Gesamtanlage ist bereit für die Abnahmetests durch den Kunden. Fehlerzustände sind dokumentiert. “Lessons Learned” ausgefüllt. Abgabe von Dokumenten, Planung, Code und Protokollen

Zur Visualisierung des Projektplans wird die Jira Roadmap verwendet. Das Projekt ist in verschiedene Abschnitte aufgeteilt, die hier auf der oberen Ebene mittels sogenannter “Epics” dargestellt werden (vgl. Gantt-Chart). Der Name jedes Epics ist als vorweggenommener Endzustand formuliert, damit auf den ersten Blick klar ist, was das Ziel ist. Jedes Epic wird auf User Stories und Aufgaben herunter gebrochen, die erledigt werden müssen für die Erreichung des (Teil-)Ziels. Ziel ist es eine Granularität zu schaffen, damit Aufgaben möglichst unabhängig voneinander bearbeitet und somit gut auf die Teammitglieder verteilt werden können.

2.3 Risiken

Risikobeschreibung	Hypothetisch / bekannt	Wahrscheinlichkeit	Maßnahme
Teammitglied bricht das Praktikum ab	bekannt	gering	Absprache mit Kunde über wegfallende Requirements
Teammitglied ist krank	bekannt	normal	Gute Dokumentation,

/ nicht verfügbar			Verteilung der Aufgaben an andere Teammitglieder
Kein Zugang zum Labor	bekannt	normal	Nutzung der Simulation zum Testen der Software
Verzug durch technische Schwierigkeiten	bekannt	normal	Technische Beratung anfragen bei Profs.

2.4 Qualitätssicherung

Um die Qualität der umgesetzten Features sicherzustellen, werden Unit- und Modultests mit der GoogleTest Suite erstellt. Vor dem Abschluss von Feature-Branches müssen alle Tests bestanden werden.

Mit den Abnahmetests wird die korrekte Funktion des Gesamtsystems aus Kundensicht getestet.

3 Problemanalyse

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.

3.1 Analyse des Kundenwunsches

Dieses Unterkapitel ist der Schritt zwischen Hardwareanalyse und dem Softwareentwurf. Was hat der Kunde an Informationen bereitgestellt, was wünscht er sich? Welche Informationen sind bekannt, welche fehlen? Deckt sich der Kundenwunsch mit den Möglichkeiten der Hardware?

Der Kunde stellt ein fertiges System zur Verfügung, das so programmiert werden soll, dass aufgelegte Werkstücke am Ende eines Förderbandes in vorgegebener Reihenfolge ankommen sollen. Zur Ermittlung der Werkstücktypen können deren Eigenschaften durch an der Anlage montierte Sensoren bestimmt werden. Werkstücke, die nicht in die Reihenfolge passen, sollen auf Rutschen aussortiert werden.

3.1.1 Stakeholder

Stakeholder	Interessen
Kunde	Produkt welches den User Stories entspricht.
Entwickler	Fertigstellung vor Deadline.
Anwender	Safety.

3.1.2 Systemkontext des Systems

Um die Kapselung Ihres Gesamtsystems klar zu dokumentieren sollten Sie ein Systemkontext erstellen. Der Systemkontext kann sich auch in einem Use Case Diagramm wiederfinden. Die Use Cases und Test Cases müssen zu der hier verwendeten Darstellung konsistent sein.

3.1.3 Anforderungen

Lfd. Nr. / ID	Beschreibung	Fußnote
ANF01	Auf der Anlage sollen die Werkstücke in folgender vorgegebener Reihenfolge sortiert werden: <Type A> → <Type B> → <Type C>	12
ANF02	Alle nicht binären Werkstücke, sollen über eine Konfigurations-Datei konfigurierbar sein.	15
ANF03	Flache Werkstücke werden von FBM1 erkannt und aussortiert, sofern sie nicht der Konfiguration entsprechen oder FBM1 voll ist.	17
ANF04	Werkstücke, die nicht der vorgegebenen Reihenfolge entsprechen, werden vom FBM2 aussortiert.	18
ANF05	Auf dem FBM1 können sich mehrere Werkstücke befinden.	22
ANF06	Auf dem FBM2 darf sich maximal 1 Werkstück befinden.	23b
ANF07	Auf der Anlage sollen die Werkstücke langsam durch die Höhenmessung transportiert werden	25
ANF08	Es darf kein Werkstück von der Anlage fallen.	26
ANF09	Sind beide Rutschen voll, läuft der Sortierbetrieb so lange weiter, bis eine Aussortierung eines Werkstückes nicht mehr erfolgen kann.	28
ANF10	Ist die Rutsche auf FBM1 voll, so soll die Aussortierung über FBM2 erfolgen	38
ANF11	Ist die Rutsche auf FBM2 voll, so soll die Aussortierung über FBM1 erfolgen.	39
ANF12	Wenn sich auf FBM1 kein Werkstück befindet, soll FBM1 anhalten.	37
ANF13	Wenn sich auf FBM2 kein Werkstück befindet, soll FBM2 anhalten.	37

ANF14	Wenn ein Werkstück das Ende von FBM 2 erreicht, werden folgende Daten auf der Konsole ausgegeben: Werkstück-ID Werkstück-Typ Höhenmesswert aus der Mitte des Werkstücks von FBM1 Höhenmesswert (höchster Wert) von FBM 2 Überschlagen (ja/nein)	29-33
ANF15	Überschlagene Werkstücke sollen auf der Konsole ausgegeben werden.	34
ANF16	Eine volle Rutsche ist an der entsprechenden Anlage zu signalisieren.	27
ANF17	Die HAL kann verschiedene Kombinationen der Weiche (Auswerfer oder Weiche) einer FBM abstrahieren.	54
ANF18	Die Lichtschranke an der Höhenmessung darf nicht im Production-Mode verwendet werden.	58

3.1.4 Use Cases / User Stories

Falls Sie Use Cases oder User Stories zur besseren Darstellung der Anforderungen erstellen wollen, dokumentieren Sie hier, welche Use Cases/ User Stories Sie auf der Systemebene implementieren müssen. Die Test Cases sollen später zu den Use Cases/ User Stories konsistent sein.

< Hier kommt die genaue Beschreibung der Use Cases. Pro Anforderung eine Tabelle benutzen (äquivalent zu SE1). Die Tabelle nach Belieben vervielfältigen. >

3.2 Hardware: Analyse der technischen Gegebenheiten

Dieses Unterkapitel soll sich mit den technischen Gegebenheiten auseinandersetzen. Mögliche Fragestellungen: Was für Hardware ist vorhanden und soll genutzt werden? Was sind Besonderheiten? Für die spätere Software nötige Maße und Eigenschaften der Hardware? Es sind nur für die Durchführung des Projektes notwendige Aspekte aufzuführen! Nutzen Sie die Möglichkeit sich mit dem System vertraut zu machen. Spielen Sie Szenarien direkt an den Festo-Anlagen durch und finden Sie so heraus, wie das System reagiert und arbeitet.

3.2.1 Technischer Aufbau und Hardwarekomponenten

Verschaffen Sie sich und dem Leser einen Überblick, um was für ein technisches System es sich handelt und wie es aufgebaut ist bzw. wie es sich zusammensetzt. Hilfreich: Skizzen, Zeichnungen, Fotos, Beschreibungen. Welche Komponenten gibt es und welche Aufgabe haben sie? Welche Schnittstellen sind existent? Vermeiden Sie jedoch zu tief ins Detail zu gehen. Dafür gibt es Datenblätter.

3.2.2 Werkstücke

Sichern Sie sich ab, indem Sie Werkstücke beschreiben und definieren. Welche Auswirkungen haben Werkstücke mit ihren Eigenschaften auf das zu entwickelnde System? Gibt es Besonderheiten?

3.2.3 Anforderungen aus dem Verhalten und technischen Besonderheiten

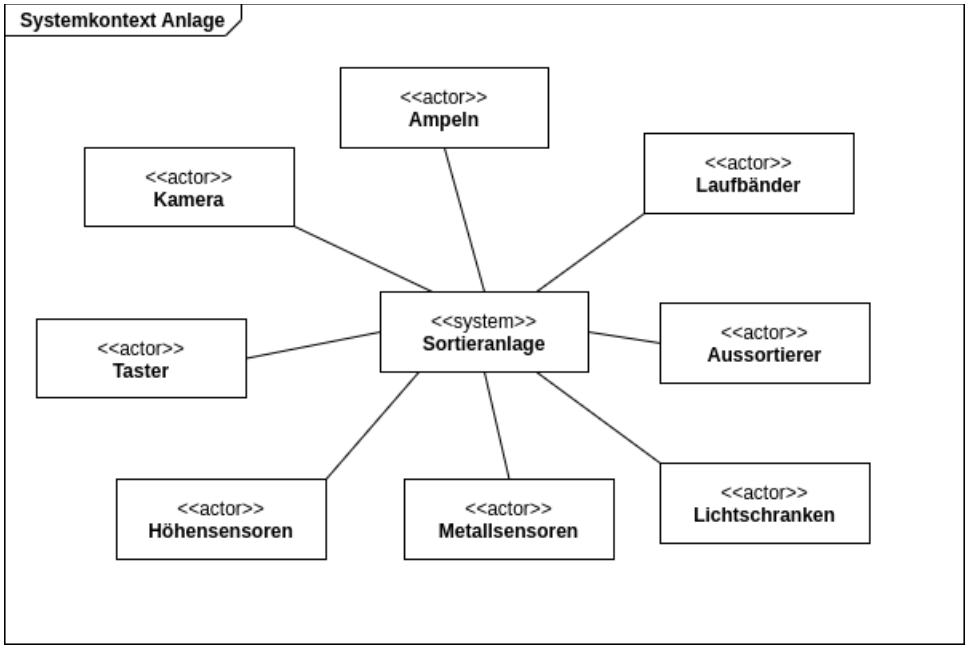
Welche Punkte aus Blickwinkel der Hardware oder der Werkstücke sind wichtig, die eventuell Aspekte der zu entwickelnden Software beeinflussen? Schauen Sie detailliert auf das Verhalten der Anlage und spielen Sie Szenarien an der Anlage durch, um Besonderheiten zu erkennen. Dokumentieren Sie ihre Ergebnisse

Lfd. Nr. / ID	Beschreibung
<HW_REQ_x>	*<Beschreibung>*

3.3 Softwareebene

Sie sollen Software für die Steuerung des technischen Systems erstellen. Aus den Anforderungen auf der Systemebene (Kundenwünsche etc.) und den Eigenschaften des technischen Prozesses (Hardware) ergeben sich Anforderungen für Ihre Software. Insbesondere wird sich eventuell die Software der beiden Anlagenteile in einigen Punkten unterscheiden. Dokumentieren Sie hier die Anforderungen, die sich speziell für die Software ergeben haben.

3.3.1 Systemkontext der Software



3.3.2 Resultierende Anforderungen an die Software

Welche wesentlichen Anforderungen ergeben sich aus den Systemanforderungen für Ihre Software? Berücksichtigen Sie auch mögliche Fehlbedienungen und Fehlverhalten des Systems. Dokumentieren Sie hier die abgeleiteten Requierements.

Lfd. Nr. / ID	Beschreibung
< REQS_x>	*<Beschreibung>*

3.3.3 Nachrichten und Signale

Welche ein- und ausgehenden Signale/Nachrichten ergeben sich aus der Hardwareanalyse und aus den Anforderungen an das System? Wie erfolgt die Kommunikation mit Nachbarsystemen? Gruppieren Sie die Nachrichten und Signale sinnvoll und teilen Sie diese in einzelne Tabellen auf.

Hinweis: Führen Sie hier schon sinnvolle und nachvollziehbare Kurzbezeichner ein, vergessen Sie aber nicht diese in den Tabellen auch entsprechend zu erläutern und zu beschreiben. Es bieten sich wiederkehrende Muster/Schemata für die Vergabe von Label und Kurzbezeichner an.

Nutzen Sie diese Bezeichner ab hier dann auch fortlaufend im Dokument und Ihrem Entwicklungsprozess!

4 Software-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.

4.1 Software Architektur

Erstellen 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 Patterns werden gegebenenfalls verwendet? Wie erfolgt die interne Kommunikation?

4.2 Software Struktur

Der nächste Schritt ist die Verfeinerung der Architektur. Zeigen Sie die Struktur ihrer Software-Komponenten und dokumentieren Sie sie 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 ihrer Struktur, also der Komponenten und ihres Aufbaus an (Klassen, Erläuterungen der Zuständigkeiten (Responsibility), Schnittstellen usw.).

4.3 Verhaltensmodellierung

Ihre Software muss zur Bearbeitung der Aufgaben ein Verhalten aufweisen/abbilden. Überlegen Sie sich dieses Verhalten auf Basis der Anforderungen und modellieren Sie das Verhalten unter Verwendung von Verhaltensdiagrammen aus den Vorlesungen.

5 Implementierung

Anmerkung: Nur wichtige Implementierungsdetails sollen hier erklärt werden. Code-Beispiele (snippets) können hier aufgelistet werden, um der Erklärung zu dienen. Welche Patterns haben Sie für Ihre Implementierung benutzt.

Anmerkung: Bitte KEINE ganzen Programme hierhin kopieren!

6 Qualitätssicherung

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

6.1 Teststrategie

Definieren Sie Zeitpunkte für die jeweiligen Teststufen in Ihrer Projektplanung. Dazu können Sie die Meilensteine zu Hilfe nehmen. Überlegen Sie, wie die Test-Architektur der jeweiligen Teststufen aussehen. Verwenden Sie Testmethoden wie z.B. Grenzwertanalyse, 100% Zustandsabdeckung, 100% Transitionsüberdeckung, Tiefensuche, Breitensuche, etc. Versuchen Sie, so gut wie möglich, Ihre Tests zu automatisieren.

6.2 Testszenarien/Abnahmetest

Leiten Sie die Abnahmebedingungen aus den Kunden-Anforderungen her. Dokumentieren Sie hier, welche Schritte für die einzelnen Abnahmetests erforderlich sind und welches Ergebnis jeweils erwartet wird (Test Cases). Abnahmetests sind Blackbox-Tests!

ID	Kurzbeschreibung	Erforderliche Eingabe	Erwartete Ausgabe	Prüfung
<AT_x>	*<Name: Filter nach kleinen Buchstaben> <optional Beschreibung>*	*<Input: A,a,C,b> <opt. Vorbedingung>*	*<Output: a,b>*	*<Leerfeld für Prüfhaken bei Abnahme>*

6.3 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 waren. 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

7 Technische Schulden

Führen Sie bekannte technische Mängel Ihres Produktes auf. Zeigen Sie dem Kunden, dass Ihnen bewusst ist, was noch an offenen Punkten existieren. Damit zeigen Sie unter anderem, dass sie den Überblick über Ihr Projekt haben. Optional: Wieviel Zeit würde eine mögliche Beseitigung von einzelnen Baustellen umfassen?

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 muss und 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

Abkürzung	Bedeutung
Anlage	FBM1 und FBM2
FBM1	Erstes Förderband Modul (vorderes)
FBM2	Zweites Förderband Modul (hinteres)

9.2 Abkürzungen

Abkürzung	Bedeutung
HAL	Hardware Abstraktion Layer.
FBM	Förderband Modul.
Production-Mode	Die Anlage sortiert die Werkstücke.

9.3 Abbildungsverzeichnis

Optional: Ein gutes Dokument beinhaltet auch ein Abbildungsverzeichnis. Wir in unserem Praktikumsumfeld benötigen es nicht zwingend.