|  |  |
| --- | --- |
|  |  |
|  | Weiterentwicklung eines Messsystems zur Bestimmung der Zusammensetzung von Altpapierproben |
|  |  |
|  |  |
|  | Further development of a measuring system for deter mining the composition of recovered paper samples |
|  |  |



|  |  |
| --- | --- |
| Kai Domhardt | 1659492 |
| Miriam Maas | 1704127 |
| Felix Knoedl | 1718621 |
| Matthias Masarczyk | 1796119 |
| Lars Emig | 2061649 |
| Michael Onufrijew | 1686368 |
| Studiengang: | M.Sc. Mechanical and Process Engineering |
|  |  |
| Advanced Design Projekt |  |
|  |  |
| Thema: | Weiterentwicklung eines Messsystems zur Bestimmung der Zusammensetzung von Altpapierproben |
|  |  |
| Eingereicht: | 10.05.2019 |
|  |  |
| Betreuer: | M.Sc. Tobias Krebs |
|  |  |
|  | Prof. Dr.-Ing. Samuel Schabel  Fachgebiet Papierfabrikation und mechanische Verfahrenstechnik Fachbereich Maschinenbau Technische Universität Darmstadt Alexanderstraße 8  64283 Darmstadt |

#### Erklärung zur vorliegenden Arbeit gemäß § 22/7bzw. § 23/7 APB

Hiermit versichern wir, die vorliegende studentische Arbeit ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, 10. Mai 2019

|  |  |
| --- | --- |
| Kai Domhardt | |
| Miriam Maas | |
| Felix Knoedl | |
| Matthias Masarczyk | |
| Lars Emig |  |
| Michael Onufrijew | |

#### Aufgabenstellung

#### Kurzfassung

In der Papierindustrie wird Altpapier als kostengünstiger und umweltfreundlicher Sekundärrohstoff für die Herstellung neuer Papierprodukte verwendet. Aufgrund der heterogenen Zusammensetzung von Altpapier und der sich daraus ergebenden unterschiedlichen Eignung für verschiedenartige Papierprodukte, wird es in Altpapiersortieranlagen aufbereitet, sortiert und zur Herstellung bereitgestellt. Am Institut für Papierfabrikation und Mechanische Verfahrenstechnik (PMV) findet im Rahmen des Forschungsprojekts AiF 19118 die Entwicklung einer automatisierten Altpapiersortieranlage statt, die durch Bildverarbeitung und Machine Learning eine Klassifikation von Altpapierobjekten vornimmt. Aktuell erfolgt die Befüllung der Altpapiersortieranlage manuell, die Vorvereinzelung wird durch eine Trommel realisiert, anschließend vereinzelt ein Industrieroboter die Altpapierobjekte und übergibt sie an die Messzelle. Ziel dieses ADPs ist die Entwicklung und Implementierung von Konzepten zur Anlagenbefüllung und Messzellenzuführung. Außerdem soll ein Graphic User Interface zur Überwachung und Steuerung der Anlage programmiert werden, das vorliegende Greifsystem des Industrieroboters analysiert und ein Greiferkonzept erarbeitet werden. Zunächst wurden die Anforderungen für jedes Teilsystem sowie das Gesamtsystem definiert. Mithilfe der Funktionssynthese und des morphologischen Kastens wurden Teillösungen erarbeitet und bewertet. Die zum Zweck der Problemanalyse der bestehenden Anlage durchgeführten Messreihen und die zuvor aufgestellte Anforderungsliste unterstützten die Auswahl der Gesamtkonzepte für jedes Teilsystem. Es wurden Anpassungen an der Struktur der Vorvereinzelungstrommel vorgenommen, um Kompatibilität mit den Gesamtkonzepten der Anlagenbefüllung zu gewährleisten, die im Anschluss auf ihre Funktionalität getestet wurden und eine Auswahl getroffen wurde. Bei der Messzellenzuführung wurde eine Anpassung des bestehenden Systems ausgewählt. Im Anschluss erfolgte die Konstruktion, Fertigung und Implementierung der Konzepte. Abschließend wurden mit einer Messreihe die erarbeiteten Lösungen validiert. Zur Optimierung des Greiferkonzeptes wurde ein Überblick über den Stand der Technik gegeben sowie Maßnahmen zur Erhöhung der Greifereffektivität aufgezeigt. Eine GUI mit grundlegenden Funktionen zur Überwachung und Steuerung der Anlage wurde implementiert.

#### Abstract

Recycled paper is being utilized as affordable and enrivonment-friendly secondary resource for the production of new paper products. Due to heterogeneous composition of recycled paper and thereby differing applicability for varied paper products it is being processed, sorted and prepared for industry by recycled paper sorting systems. The Institut für Papierfabrikation und Mechanische Verfahrenstechnik (PMV) is conducting the research project AiF 19118 which aims at developing a fully automated recycled paper sorting system, using image processing and machine learning in order to perform classification of recycled paper objects. Currently the sorting system’s filling process is performed manually, the pre-separation is accomplished by a drum screen, subsequently an industrial robot separates the paper objects and transfers them into the measurement cell. This Advanced Design Project aims at developing and implementing a filling system as well as a measurement cell feed. Additionally, a Graphic User Interface for constant monitoring and controlling the sorting system is to be programmed, the current industrial robot’s gripping concept is to be analyzed and an improved gripping concept to be developed. At first requirements for every subsystem as well as for the overall system were defined. Using the techniques of function synthesis and morphological boxes partial solutions were compiled and evaluated. Both the requirements list and a problem analysis resulting from a series of tests supported the overall concept selection process for every subsystem in question. Structural adjustments to the drum screen were conducted in order to ensure compatibility and allow to further test and evaluate overall filling system concepts. Regarding the measurement cell feed the existing system was chosen to be improved on. Following this, the subsystem concepts were designed in detail, manufactured and implemented. Subsequently, a final series of tests was conducted to validate the selected solutions. For the gripping system a state of the art technology overview was compiled and a set of recommended measures to improve the existing gripping effectivity was presented. A GUI with basic monitoring and control features for the overall system was implemented.

# Inhaltsverzeichnis

1. Inhaltsverzeichnis viii

2. Einleitung 1

2.1. Motivation 1

2.2. Zielsetzung (Michael) 1

2.3. Aufbau der Arbeit (Miriam) 2

3. Methodik der Vorgehensweise (Felix) 3

3.1. Einarbeitung in die Thematik 3

3.2. Projektdefinitionsprozess 3

3.3. Konzeptprozess 3

3.4. Entwurfs- und Konstruktionsprozess 3

4. Stand der Technik 3

4.1. Beschreibung der bisherigen Anlage 4

4.1.1. Anlagenbefüllung 4

4.1.2. Vorvereinzelung 4

4.1.3. Vereinzelung 5

4.1.4. Messzelle 6

4.2. Aufbau moderner Altpapiersortieranlagen 8

4.3. Greifprinzipien für biegeschlaffe Materialien 12

4.4. Graphische Oberfläche (GUI) (Kai) 15

5. Projektdefinition 16

5.1. Zeitmanagement 16

5.2. Aufgabenklärung 16

5.3. Systemanalyse des Ausgangszustands 17

5.3.1. Versuchsdurchführung der Messreihen 17

5.3.2. Auswertung der Messreihen 18

5.3.3. Problem- und Ursachenanalyse (Matthias) 20

5.4. Anforderungsliste Gesamt-/Teilsysteme (Matthias) 21

6. Konzeption 22

6.1. Funktionsmodelle aufstellen (Felix) 22

6.2. Morphologischer Kasten (Michael) 22

6.3. Bewertung der Teillösungen (Michael) 23

6.4. Generierung von Konzepten aus dem morphologischen Kasten 25

6.4.1. Lösungskonzepte für die Anlagenbefüllung 26

6.4.2. Lösungskonzepte für die Messzellenzuführung 30

6.5. Beurteilung der Konzepte (vor Neukonstruktion der Vorvereinzelung) (Miriam) 34

6.5.1. Konzeptauswahl mit FF und BF 34

6.5.2. Konzeptbewertung ZF und W 34

6.6. Umstrukturierung der Vorvereinzelung (Lars) 34

7. Detaillierung und Konstruktion 34

7.1. Detaillierung der Konzepte (Lars) 34

7.2. Konstruktive Anpassung der Trommel (Lars) 34

7.3. Konstruktion Befüllanlage (Lars) 34

7.4. Konstruktion Messzellenzuführung (Felix) 34

8. Greifer (Matthias) 35

9. Software (Kai) 36

9.1. Programmablaufplan des IST- Zustands 36

9.2. Programmoptimierung (Ablauf, Prozess, Fehler) 36

9.3. GUI 36

10. Inbetriebnahme und Validierung 36

11. Fazit 36

11.1. Zusammenfassung 36

11.2. Ausblick 36

12. Anhang 1

Abbildungsverzeichnis 1

Abkürzungsverzeichnis 1

Literaturverzeichnis 1

Indexverzeichnis 2

# Einleitung

## Motivation

# Methodik der Vorgehensweise (Felix)

## Einarbeitung in die Thematik

## Projektdefinitionsprozess

## Konzeptprozess

## Entwurfs- und Konstruktionsprozess

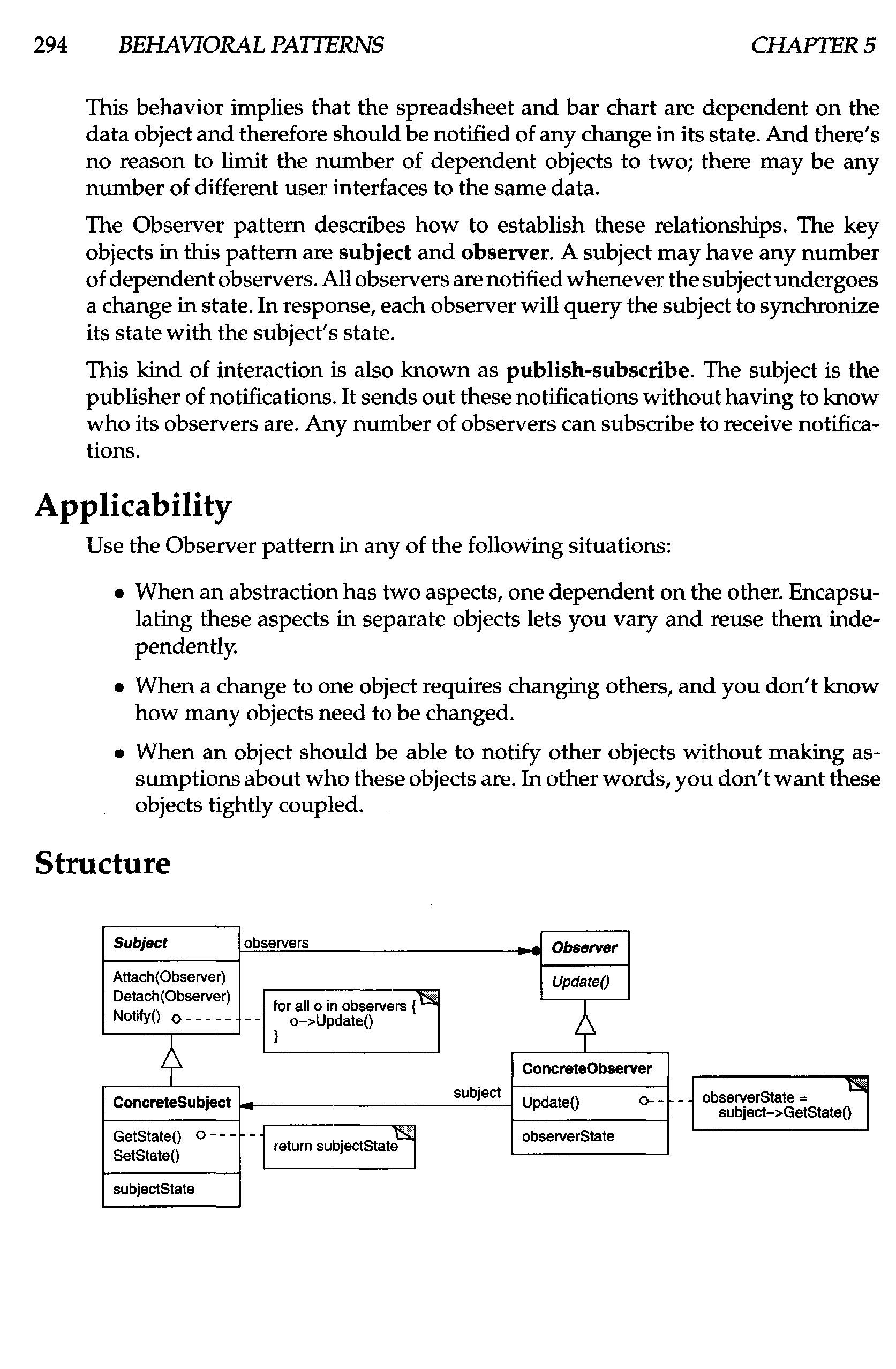
# Stand der Technik

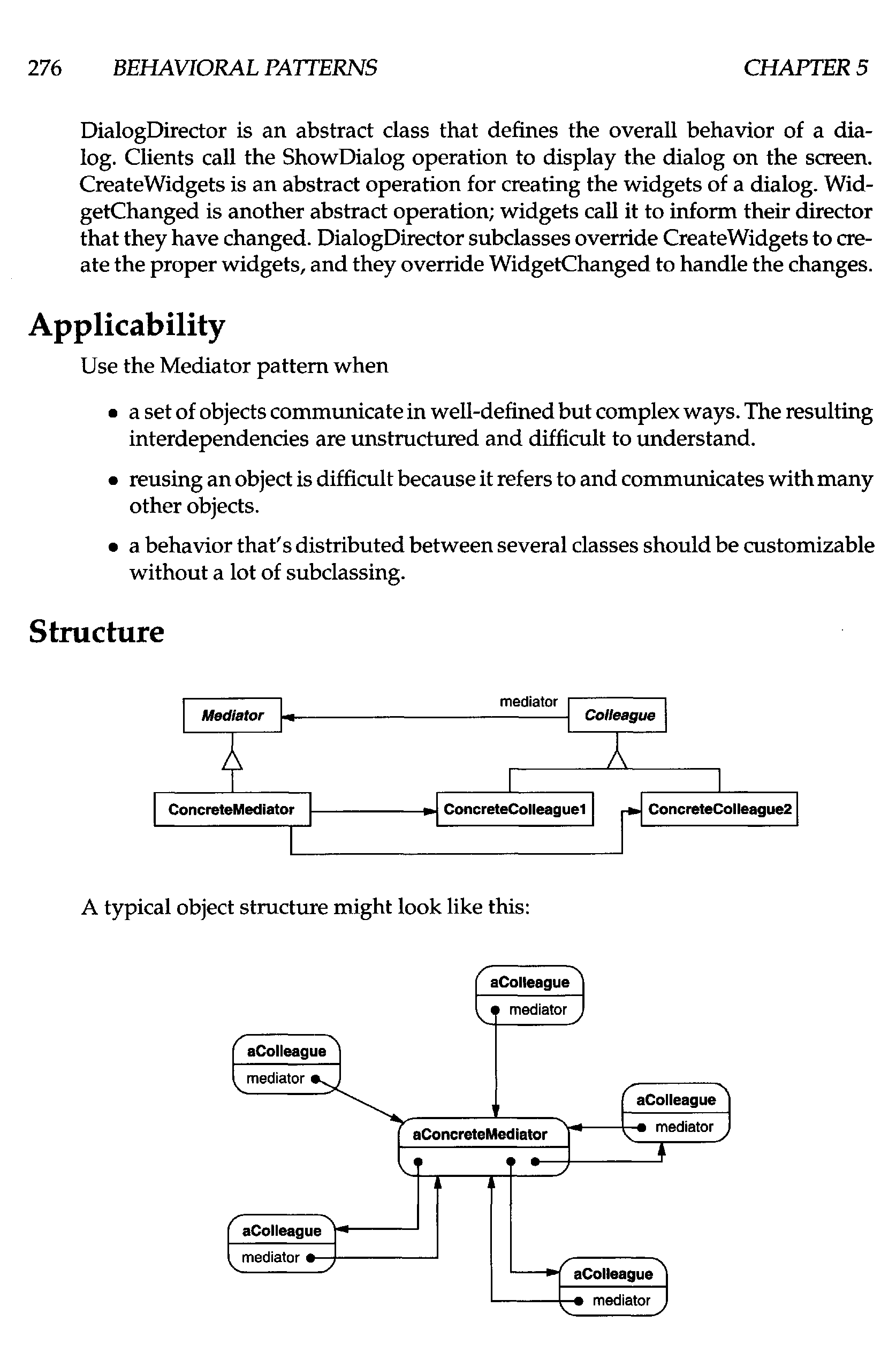
## Graphische Oberfläche (GUI) (Kai)

In Bezug auf die Gestaltung einer grafischen Oberfläche spielt der Begriff Usability eine bedeutende Rolle (Richter & Flückiger, 2013). Wörtlich übersetzt kann unter diesem Begriff die Benutzerfreundlichkeit bzw. die Gebrauchstauglichkeit verstanden werden (Schlick et al., 2010). Außerdem spielen Aspekte wie die Zufriedenstellung des Nutzers sowie seine Akzeptanz in diese Begrifflichkeit mit ein (Nielsen, 2010). Für eine zufriedenstellende Usability existieren verschiedene Faktoren, welche nachfolgend kurz erläutert werden (Nielsen, 2010):

1. **Einfach zu erlernen:** Der Anwender gewinnt leicht ein Verständnis von dem System und kann hiermit umgehen.
2. **Effiziente Nutzbarkeit:** Der Anwender kann nach einer kurzen Einarbeitungsphase effizient mit dem System arbeiten.
3. **Wenige Fehlermöglichkeiten:** Das System zeichnet sich durch eine geringe Anzahl von Fehlermöglichkeiten aus und unterlaufene Fehler können schnell revidiert werden.
4. **Individuell und angenehm:** Das System sollte individuell angepasst werden können und ein angenehmen Umgang gewährleisten.
5. **Einfach sich zu erinnern:** Der Benutzer sollte nach einer langen Phase der Nicht Benutzung schnell wieder in Benutzung des Systems zurückfinden und sich einfach erinnern.

Bei der konkreten Umsetzung werden, erprobte halbstandartisierte Systemstrukturen, sogenannte Entwurfsmuster verwendet. Für grafische Oberflächen werden besonders die Entwurfsmuster „Beobachter“ (ZITAT Erich Gamma, et al, Design Pattern Elements of Reusable Object Oriented Software S. 293ff)(engl. Observer), sowie „Vermittler“ (ZITAT Erich Gamma, et al, Design Pattern Elements of Reusable Object Oriented Software S. 273ff) (engl. Mediator) eingesetzt.

  
Abbildung 4-1: Struktur Entwurfsmuster „Beobachter" (ZITAT Erich Gamma, et al, Design Pattern Elements of Reusable Object Oriented Software S. 294)

  
Abbildung 4-2: Struktur Vermittler (ZITAT Erich Gamma, et al, Design Pattern Elements of Reusable Object Oriented Software S. 276)

Das Entwurfsmuster Beobachter dient zu Bedarfsorientierten Erneuerung der in der Oberfläche dargestellten Daten.

# Projektdefinition

# Konzeption

# Detaillierung und Konstruktion

# Software (Kai)

In diesem Abschnitt wird die Entwicklung einer grafischen Nutzeroberfläche (engl. Graphical User Interface – GUI) zur einfacheren Prozesssteuerung, sowie –überwachung dokumentiert. Hierzu wurde in einem ersten Schritt das bestehende System betrachtet und analysiert. Hierzu werden alle Klassen die von Nutzern für Erweiterungen (Einbau neuer Sensorik etc.) modifiziert werden müssen, als „User space“ bezeichnet. Im Folgenden werden alle, mit der GUI zusammenhängenden, Neuentwicklungen als dem „Library space“ zugehörig betrachtet und dürfen somit keine direkte Modifikation seitens des Nutzers bei der Prozesssteuerung oder Prozesserweiterung erfordern. Diese Unterteilung dient dazu, die Möglichkeit der unbewussten Einbringung von Fehler in Klassenübergreifenden Funktionen einzudämmen, bzw. zu verhindern.

## Ausgangzustand

Das System wie es zum Ausgangszeitpunkt dieses Projektes besteht beschränkt sich ausschließlich auf den User space. In Abbildung 8-1 ist das Gesamtsystem abgebildet, sämtliche für die Anlage entwickelten Klassen, ab hier als Anlagenklassen bezeichnet, sind explizit als solche dargestellt. Des Weiteren sind alle Attribute, welche für Kommunikationsschnittstellen zur physikalischen Anlage genutzt werden und selbst keine Anlagenklasse sind, in der jeweiligen Klasse aufgelistet.

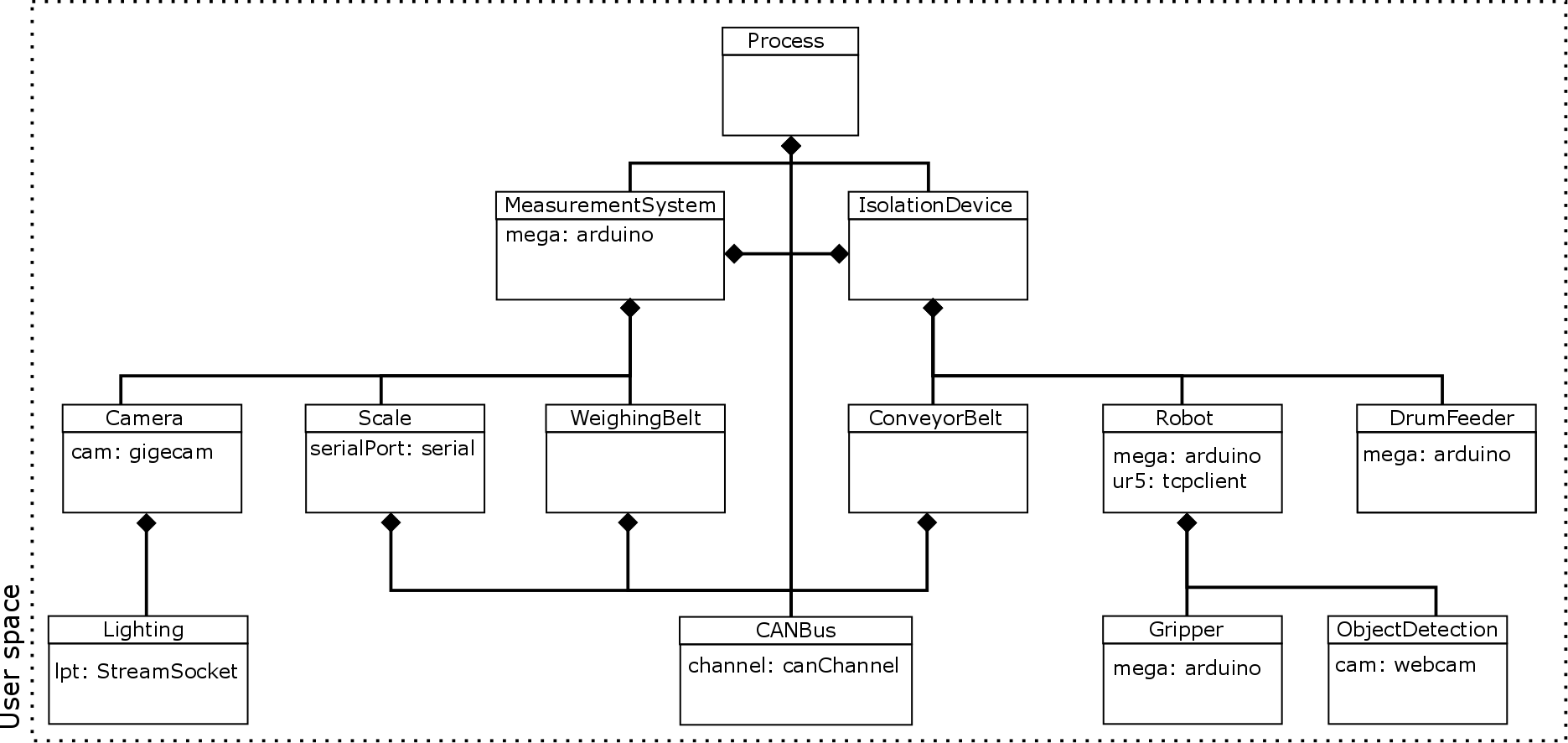


Abbildung 8-1 Klassendiagramm für Ausgangssituation

Die Steuerung des Prozesses, erfolgt ausschließlich über Methodenaufrufe in die Matlabkonsole. Dies erfordert, dass der Nutzer mit der Softwareimplementation des Prozesses vertraut ist um sich in der Klassenstruktur zurecht zu finden. Ebenso ist es Notwendig zu wissen, welches Verhalten durch welche Methode hervorgerufen wird.

Das bisherige Überwachungskonzept stützt sich auf drei Methoden zur Bestimmung des Anlangenzustands. Primär kann die Anlage über Interaktion mit den Instanzen der Anlagenklassen überwacht werden. Dieser Ansatz erfordert aber auch, wie schon die Steuerung, dass der Nutzer tieferes Wissen über die Art der Implementierung besitzt. Sekundär werden über ein rudimentäres Loggingsystem Zustandsnachrichten in der Matlabkonsole wiedergegeben. Dieses Konzepte birgt einige Problempunkte. So ist das Überwachungsprotokoll nicht Persistent und beschränkt sich auf die Dauer in der die Instanz der ausführenden Matlabumgebung aktiv ist. Ebenso ist es nicht Filterbar, welches das Überwachen jeder Zustandsnachricht erfordert, auch wenn nur ein bestimmtes Subset von Interesse ist.

Tertiär wird die Anlage durch den Anlagenbenutzer visuell überwacht. Dies erfordert die permanente Anwesenheit des Nutzers, wodurch das Betreiben der Anlage sehr Aufwands intensiv ist. Ebenso können die internen Zustände der Objektrepräsentationen und der Komponentensoftware selbst nicht direkt überwacht werden. Es ist somit erst das Auftreten eines Fehlers feststellbar, jedoch nicht das Auftreten seiner Ursache. Dies erschwert das Nachvollziehen zum Beheben eines Fehlers.

## Neuentwicklung

Um den bisherigen Stand zu verbessern und die zuvor beschriebenen Probleme zu beheben, wurde ein neues Steuer- und Überwachungskonzept implementiert. Zur einfachen Bedienung ist dieses in einer interaktiven grafischen Nutzeroberfäche umgesetzt. Dieses neue Konzept besteht aus drei Elementen mit unterschiedlichen Funktionen und Detaillierungsgraden. Ein Klassendiagramm, welches die Struktur sämtlicher Neuentwicklungen darstellt ist im Anhang zu finden.

### Zustandsanzeige

Das oberste Element bildet die Zustandsanzeige, wie sie in der GUI Hauptansicht, zu sehen in Abbildung, verwendet wird. Hier wird der Zustand jeder Anlagenklasse über einen Farbcode, sowie einem Kurztext zur genaueren Beschreibung dargestellt. Somit hat der Nutzer die Möglichkeit, sich mit einem Blick einen Eindruck zum Gesamtzustand der Anlage zu verschaffen.

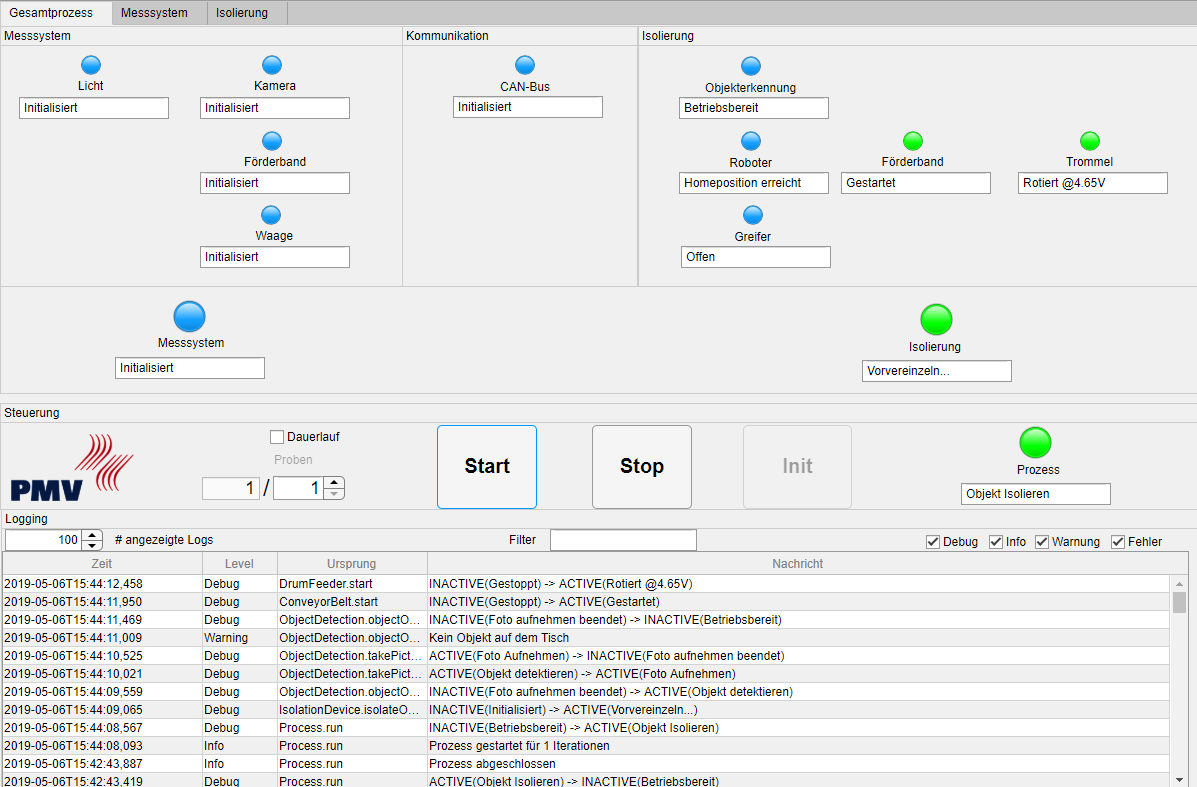


Abbildung 8-2 Hauptansicht GUI

Um die Zustandsmechanik zu realisieren wurde eine StateObject Klasse implementiert, deren Eigenschaften alle Anlagenklassen erben. In dieser Systematik, kann jedes Objekt einen der sechs in Tabelle beschriebenen Zuständen haben.

Tabelle 1 Zustände von Anlagenklassen

|  |  |  |
| --- | --- | --- |
| **ID** | **Zustand** | **Beschreibung** |
| 1 | Offline | Die Instanz der Anlagenklasse wurde erstellt, es wurde aber noch keine Verbindung zur Anlage initialisiert |
| 2 | Inaktiv | Das Objekt führt derzeit keine Funktionen aus, und hat keine Funktionen an Subobjekte delegiert |
| 3 | Aktiv | Das Objekt führt derzeit eine Funktionen aus, oder hat eine Funktionen an Subobjekte delegiert |
| 4 | Unbekannt | Der Zustand der Anlage kann nicht überprüft werden (z.B. bei Verbindungungsproblemen) |
| 5 | Gestört | Die Funktion des Objektes kann nicht weiter ausgeführt werden. Dieser Zustand blockiert die Ausführung weitere Aktionsmethoden (siehe KAPITEL X) |
| 6 | Gestoppt | Die Funktion des Objektes wird durch den Nutzer blockiert. |

Die Zustände mit den erlaubten Übergängen sind in Abbildung zu sehen. Blaue Pfeile markieren Übergänge, welche prinzipiell möglich sind, jedoch nur vom Nutzer initiiert werden sollten. An dieser Stelle soll besonders darauf hingewiesen werden, dass die Zustände Gestört, sowie Gestoppt nur über einen nutzerseitigen Zustandswechsel angedacht sind. Nach einer Fehlerbehebung durch den Nutzer, sollte der die Anlage in den Zustand Unbekannt und durch einen Selbsttest (zu implementieren in der Methode updateState()) in die den Zustand Inaktiv oder Gestört überführt werden..

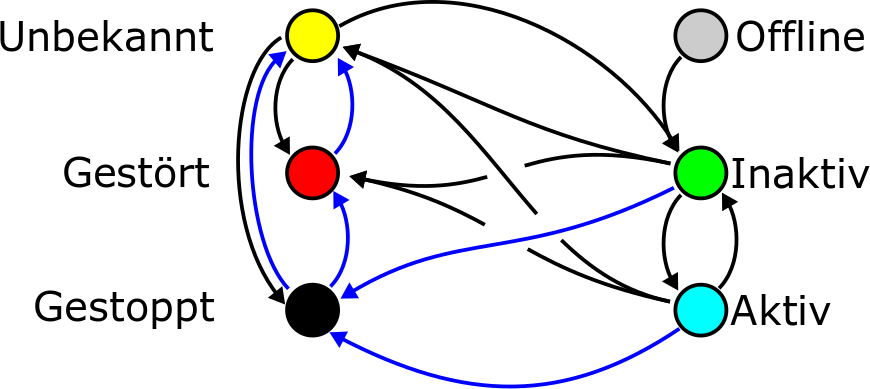


Abbildung 8-3 Zustandsübergänge

Damit eine Anlagenklassen diese Zustandsfunktionalität nutzen kann, müssen eine Reihe an Kriterien erfüllt sein. Die Anlagenklasse muss:

1. Eine Subklasse von StateObject sein.
2. Die Methode updateState() implementieren. Diese Methode definiert wie die Anlagenklasse ihren eigenen Zustand bestimmen kann.
3. Die Methode onStateChange() implementieren. Diese Methode definiert Aktionen die bei einem Zustandsübergang getätigt werden sollen, wie z.B. automatische detailierte Protokollierung bei dem Zustand Gestört.

Nicht erforderlich, jedoch optional um den vollen Funktionsumfang nutzen zu können, ist die Deklaration von Aktionsmethoden, wie sie in Abbildung zu sehen ist. Hierbei handelt es sich um Methoden, welche nicht nur sensorische, oder planerische Funktionen beinhalten, sonder Aktionen ausführen. Diese Deklaration bewirkt, dass von der jeweiligen Anlagenklasse keine Signale an die Anlage gesendet werden, sollte sich das Objekt im Zustand Gestört oder Gestoppt befinden. Ein Beispiel für die blockierende Funktion von Aktionsmethoden ist in Abbildung dargestellt.

|  |
| --- |
| function actionMethod(this,varargin)  if ~this.isReady; return; end  ... |

Abbildung 8-4 Matlab Code zur Deklaration von „Aktionsmethoden“ über Aufruf der isReady Methode

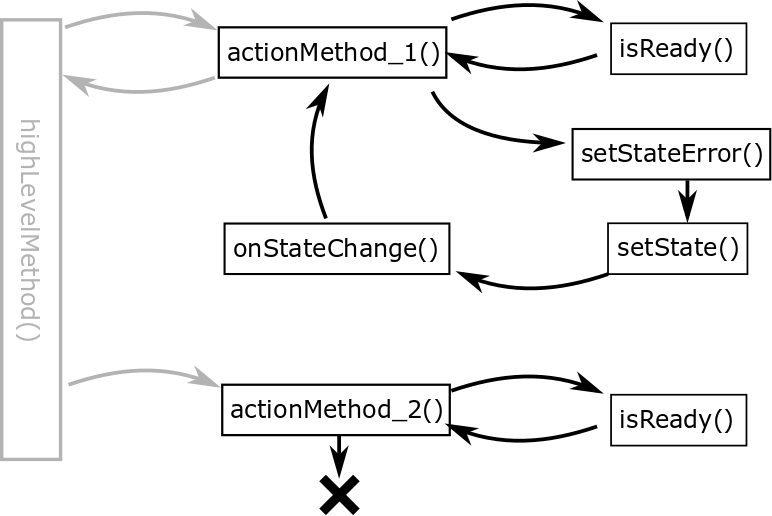


Abbildung 8-5 Selbstblockade bei „Aktionsmethoden"

### Logging

Der Zustandsanzeige nachgestellt ist das Loggingsystem, welches Nachrichten von den Anlagenklassen dem Nutzer zugänglich macht. Die in Abbildung dargestellte Struktur dieses Systems besteht aus einem Logger Objekt, welches die Nachrichten verwaltet, und einer Anzahl an LogEntry Objekten, welche die system-interne Repräsentation der Nachrichten sind. Jedes Objekt mit Zugang zu dem Logger Objekt kann eine Log-Nachricht durch den Logger erstellen lassen. Hierbei wird in dem LogEntry Objekt, die Nachtricht, der aktuelle Zeitstempel, das Level der Nachricht (Hier: Debug, Info, Warning, Error), sowie der Call-Stack ZITAT (https://de.mathworks.com/help/matlab/ref/dbstack.html) der initiierenden Funktion gespeichert.

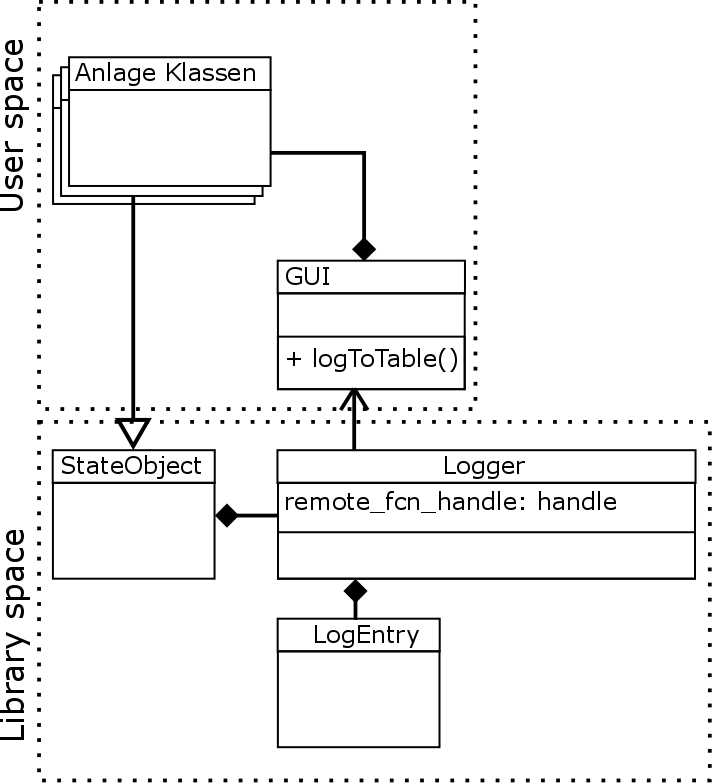


Abbildung 8-6 Klassendiagramm Logger

Die Umsetzung der Visualisierung des Logging Systems kann aus Abbildung entnommen werden. Hier ist auch zu sehen, dass nach den drei Eigenschaften einer Nachricht (Zeit, Level und Ursprung) gefiltert werden kann. Die Einstellungen für das Logging System sind in Abbildung dargestellt.

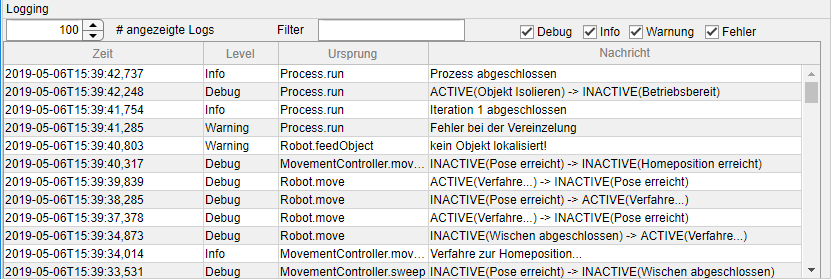


Abbildung 8-7 Wiedergabe von Statusnachrichten in der GUI

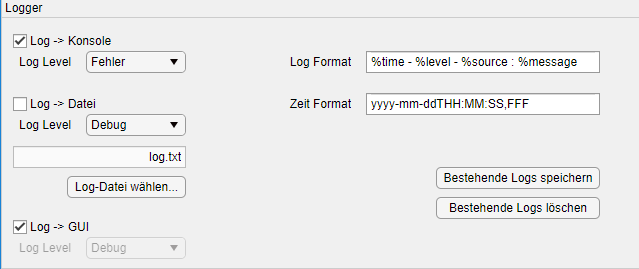


Abbildung 8-8 Einstellungsmöglichkeiten Logger

### Detailanzeige

Die dritte Ebene bilden die Detailanzeigen, hier werden konkrete Zustandsgrößen, wie Winkel und Spannungen) der einzelnen Anlagenklassen visualisiert. Diese sind durch eine Kombination der Entwurfsmuster „Beobachter“ und „Vermittler“ realisiert, um den Prozess unabhängig von der grafischen Nutzeroberfläche verwenden zu können. Aus der in Abbildung geht die Struktur für die Detailanzeige hervor.

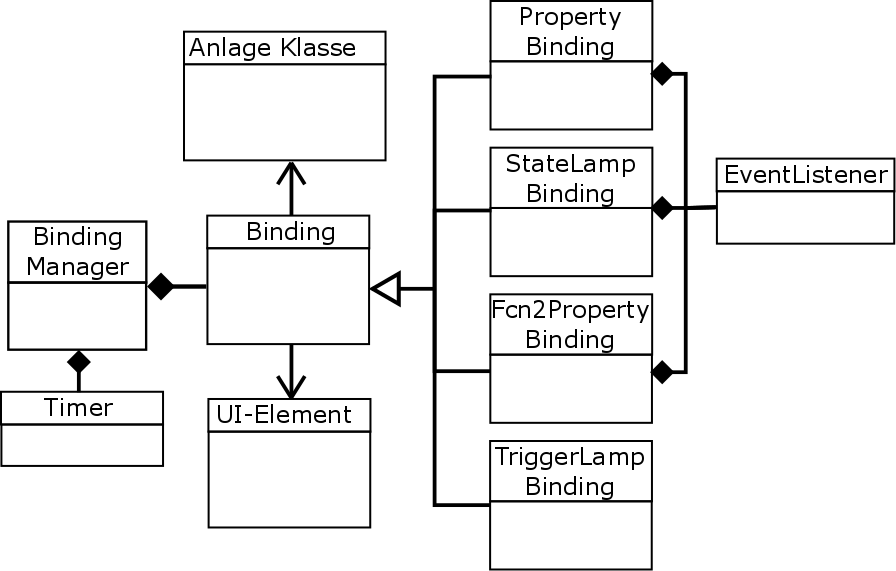


Abbildung 8-9 Klassendiagram Bindings

Die Klasse „Binding“ beobachtet, mittels eines EventListeners, einzelne Attribute[[1]](#footnote-2) der Anlagenklassen.

Damit ist die Klasse „Binding“ sowohl „Beobachter“, als auch „Vermittler“. Bei jeder Änderung des Attributes wird die Übertragungsfunktion, implementiert in den Kinderklassen, ausgewertet. Diese überführt den Wert des Attributes auf die darzustellende Form. Die Arten von bereits Implementierten Subklassen, bzw. Übertragungsfunktionen, und eine rudimentäre Beschreibung dieser, ist aus Tabelle zu entnehmen.

Tabelle 2 Subklassen von Binding

|  |  |
| --- | --- |
| **Binding Subklasse** | **Beschreibung** |
| PropertyBinding | Überträgt den Wert des Attributes des Anlagenklassens auf ein Attribut des Visualisierungselements |
| StateLampBinding | Überträgt den Zustand und dessen Beschreibung einer StateObject Subklasse auf einen Farbcode und ein Textfeld. |
| TriggerLampBinding | Überträgt den boolschen Wert eines Attributes auf einen Farbcode, für aktiv oder inaktiv. |
| Fcn2PropertyBinding | In der Übertragungsfunktion identisch mit PropertyBinding. Es wird kein Attribut überwacht, sondern eine Funktion zur Wertbestimmung verwendet. Dieses Binding kann daher nur mit einem Zeit-, jedoch nicht Ereignisgesteuert werden. |

Das derzeitige Layout der Detailanzeige für das Untersystem Isolierung kann aus Abbildung und Abbildung, sowie für das Messsystem aus Abbildung und Abbildung entnommen werden.

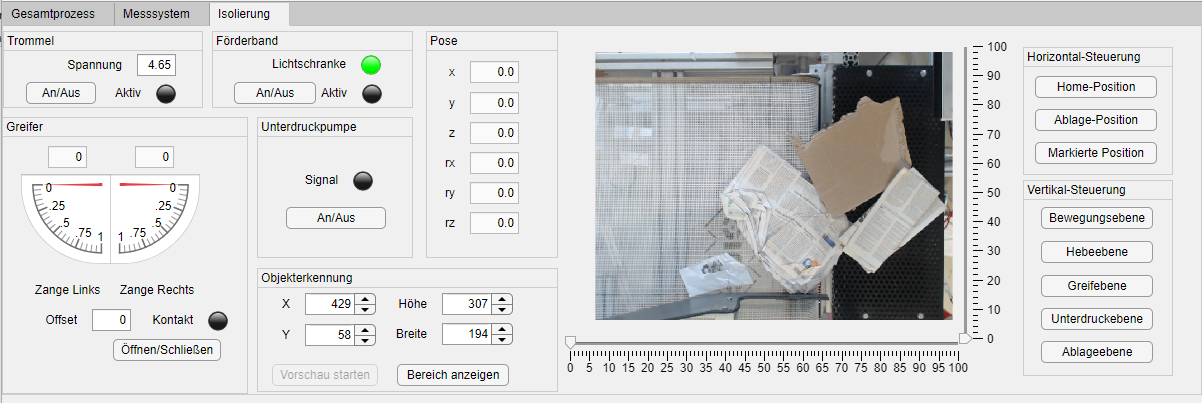


Abbildung 8-10 Ansicht Isolierung (links)

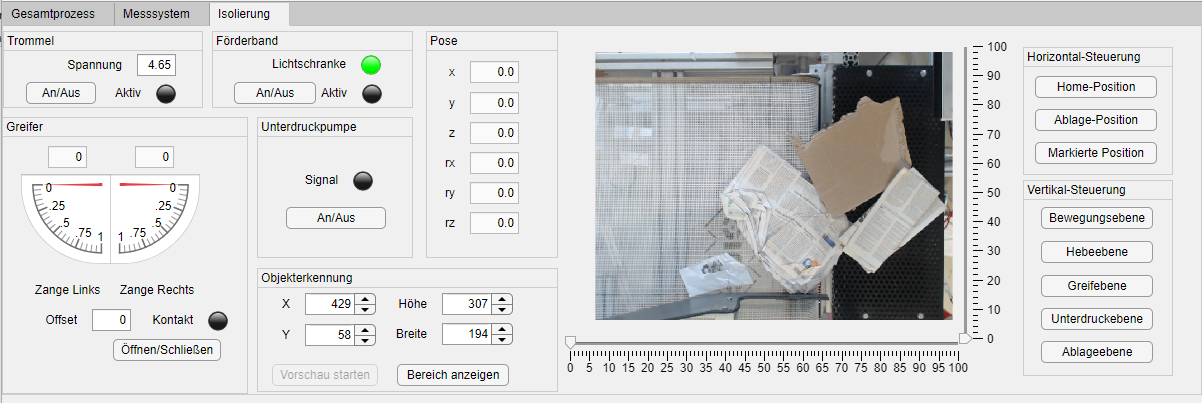


Abbildung 8-11 Ansicht Isolierung (rechts)

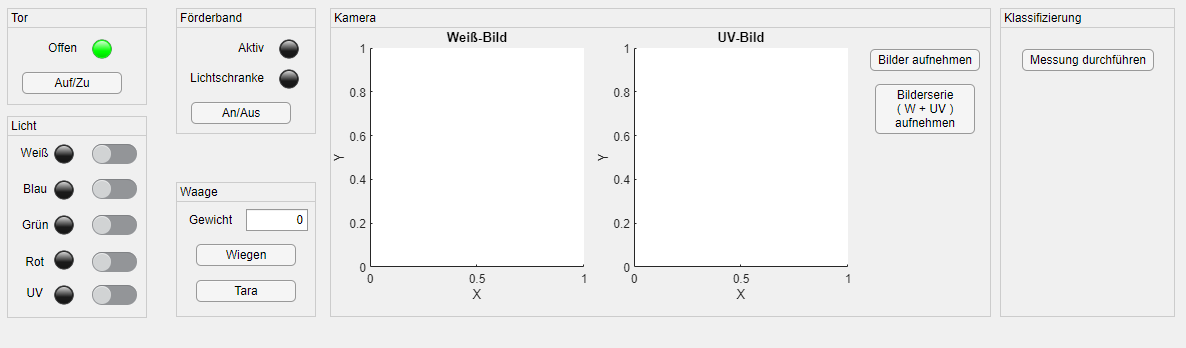


Abbildung 8-12 Ansicht Messzelle (links)

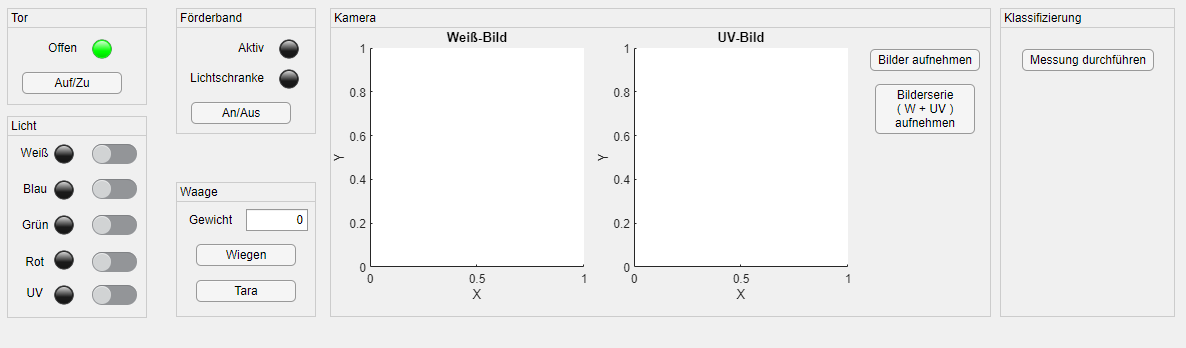


Abbildung 8-13 Ansicht Messzelle (rechts)

## Exemplarische Prozessverbesserung am Beispiel Roboter

Parallel zur Entwicklung der GUI wurde auch der bestehende Prozess analysiert. Hierbei viel auf, dass der der Prozess zum Anfangsstand des Projektes einige Altlasten aus früheren Entwicklungen beinhaltet. Beispielsweise soll hier eine Prozessverbesserung an der Ansteuerung am Roboterarm vorgenommen werden, um die Möglichkeit der Optimierung aufzuzeigen.

### IST-Zustand

Die zentrale Aufgabe des Roboterarms ist es, Altpapierobjekte von einer Ablagefläche in die Messzellenzuführung zu fördern. Diese Funktion wird in der Methode „feedObject“ realisiert. Der IST-Zustand, ist in Abbildung, sowie für Subfunktionen im Anhang, dargestellt. Hier fällt auf das die beiden beiden Aufnahmemechanismen, Greifen und Unterdruck, in separaten Aktivitätssträngen verwendet werden. Diese Aktivitätstrennung wird auch bis zum Ende der „feedObject“ Methode nicht wieder zusammengeführt, obwohl sie großteils funktional identisch sind. Beispielsweise enthalten die Methoden „moveObject“ und „moveObjectGripper“ die gleichen abzufahrenden Trajektorien.

Durch diese Trennung von ähnlichen, bzw. identischen Funktionen, ist es erschwert Änderungen konsistent in den Prozess einzubringen. Dieses Problem kann an dem fehlenden Startbefehl für das Messzellenförderband in der Methode „moveObjectGripper“, welcher jedoch in der Methode „moveObject“ vorhanden ist, nachempfunden werden.

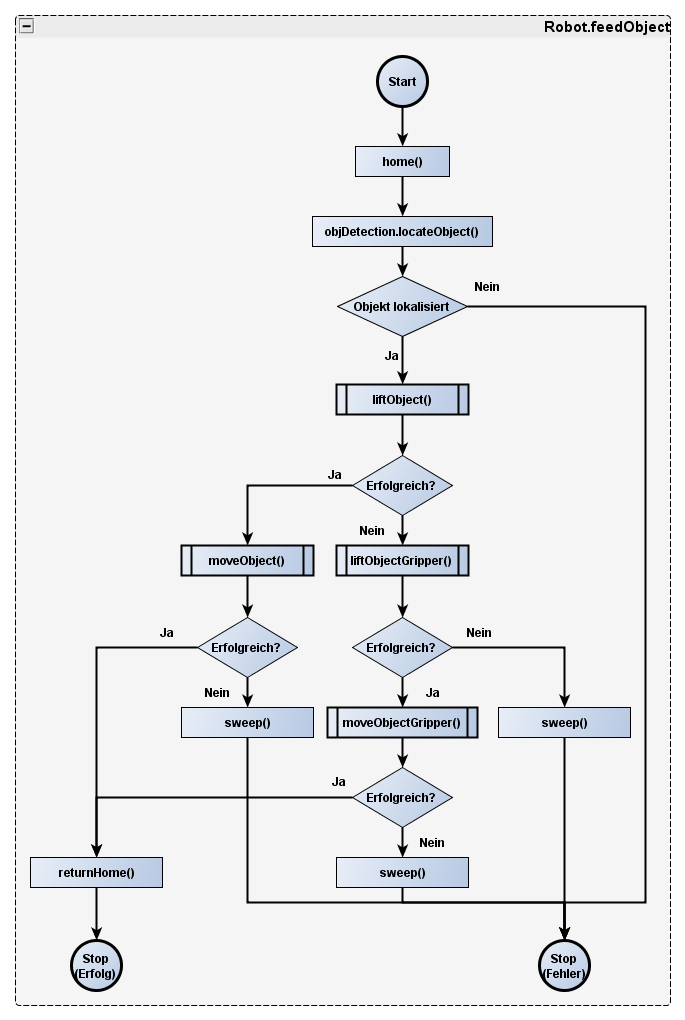


Abbildung 8-14 alter Prozess für Objektzuführung durch Roboter

### Neuer Zustand

Um den Prozess zu optimieren, wurden die bestehenden Methoden entsprechend ihrer Anwendungsfälle in eine neue Struktur, dargestellt in Abbildung, aufgeteilt.

„liftObjectGripper“ wurde in „liftObject“ integriert und die Unterscheidung der Aufnahmemechanismen wird in den Subfunktionen „pickUpVacuum“ und „pickUpGripper“ getroffen. Nach der Ausführung von „liftObject“ ist der Prozessablauf unabhängig davon, mit welchem Mechanismus die Altpapierprobe aufgenommen wurde.

Die Methode „moveObjectGripper“ wurde, analog zu „liftObjectGripper“, in „moveObject“ integriert. Hier wurde die Gelegenheit genutzt um die Trennung der beiden Anlagensysteme „MeasuringSystem“ und „IsolationDevice“ konsequent durchzusetzen. Weiterhin wurden Bewegungsbefehle durch Befehle, für das in 8.4 beschriebene Bewegungskonzept, ersetzt. Anstatt dass, wie zuvor der Roboter (zu „IsolationDevice“ zugehörig) einen direkten Befehl an das Förderband der Klasse „MeasuringSystem“ ausführt, wird nun eine indirektes System mit dem Entwurfsmuster „Vermittler“ verwendet. Hier vermittelt die oberste Klasse „Process“ die Aufforderung zur Übernahme des Altpapierobjektes zwischen Roboter und Messsystem. Initiiert wird die Übernahme durch das Ereignis „Handoff\_Request“ seitens des Roboters. Hierdurch sind „MeasuringSystem“ und „IsolationDevice“ unabhängig voneinander einsetzbar und erfordern nicht die Instanziierung des jeweils anderen Systems.

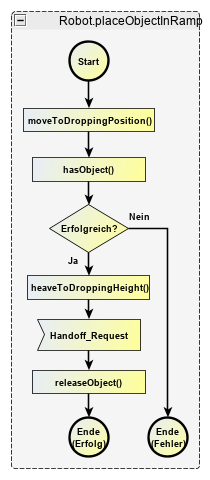
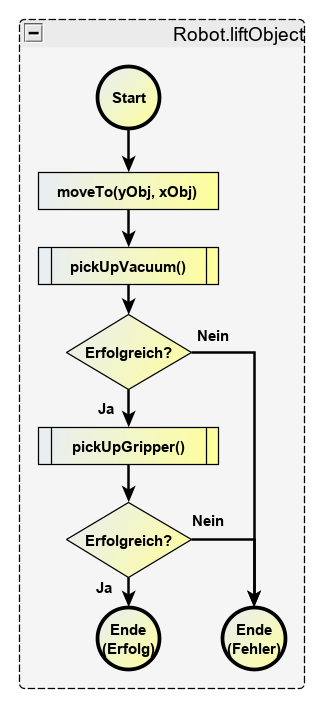
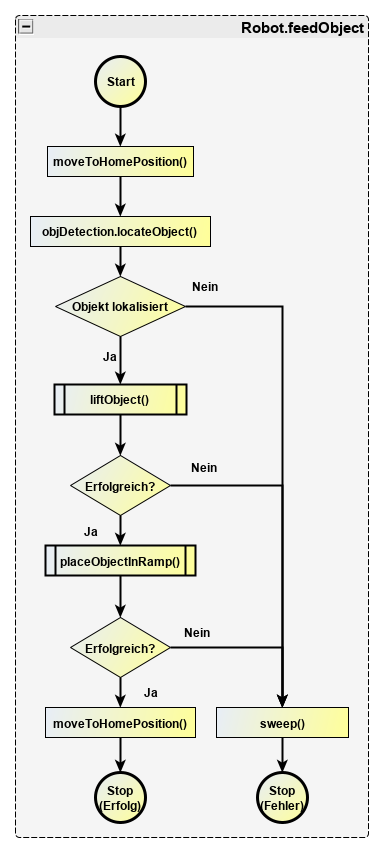


Abbildung 8-15 neuer Prozess für Objektzuführung durch Roboter

## Bewegungskonzept Roboter

Um zukünftig Vereinzelungskonzept zum Aufnehmen von Altpapierprobem validieren zu können, ist es wichtig diese schon möglichst früh direkt am Roboterarm erproben zu können. Hierzu muss dem Anwender eine Schnittstelle zur Verfügung gestellt werden, welche einfach zu erlernen und erweitern, sowie gegenüber eventueller Fehlbenutzung abgesichert ist. Letztere Anforderung trägt besonderes Gewicht, da die Entwicklung von neuen Konzepten zur Vereinzelung in erster Instanz keine tiefer gehende Kenntnis der Robotersteuerung erfordert und es bei einer hohen Entwicklungsgeschwindigkeit einfach zu Bedienungsfehlern kommen kann.

Die aktuelle Schnittstelle zum Roboterarm bietet rein die Angabe eines 1x6 Vektors, bestehen aus den Elementen X, Y, Z, RX, RY und RZ. Dieser stellt die gewünschte Endpose dar. Durch diese direkt ander Steuerung des Roboters angesiedelte Schnittstelle ist die Anforderung, dass diese einfach zu erlernen sein soll nicht erfüllt, da der Nutzer sich eine große Anzahl an Koordinaten, sowie deren Abhängigkeiten untereinander merken muss. Durch die Abhängigkeit der Koordinaten erschwert sich ebenso die sichere Benutzung des Roboterarms für ungeübte Nutzer. Die Änderung einer Koordinate kann dazu führen, dass sich alle der restlichen fünf Koordinaten ebenso ändern müssen um die Erreichbarkeit der Endpose zu garantieren. Besonders wird dieses Problem beim Verfahren von längeren Distanzen deutlich. Durch den Roboterarm wird nicht sicher gestellt, dass jede der in der Trajektorie enthaltenen Posen tatsächlich erreichbar sind und nicht zu Kollisionen mit sich selbst oder der Umgebung führen.

Um diesen Anforderung zu genügen, wurde ein neues Bewegungsmodell entwickelt, welches auf der bisherigen Schnittstelle aufbaut. In diesem Modell, wie es in Abbildung zu sehen ist, werden sechs Ebenen definiert sowie drei Achsen definiert. Jede der Ebenen hat zwei Variablen (X und Y), und vier feste Parameter (Z, RX, RY, RZ), siehe Tabelle, und ist in ihrer Ausbreitung so bemessen, dass es bei keiner Kombination von der Variablen es zu einer unerreichbaren Pose kommt. Eine horizontales Verfahren ist ausschließlich auf der Bewegungsebene Hoch sowie auf der Bewegungsebene Tief möglich. Jede weitere Pose auf einer anderen Ebene kann nur durch vertikales Verfahren von einer der beiden Bewegungsebenen aus erreicht werden. Verfährt der Roboterarm von einer Bewegungsebene zur anderen, geschieht dies über zwei Hilfsposen, welche an den Durchstoßpunkten der Wechselachse und den beiden Bewegungsebenen liegen. Ein direktes Verfahren zwischen der Greifpose und der Ablagepose würde sonst eine nicht erreichbare Pose enthalten.

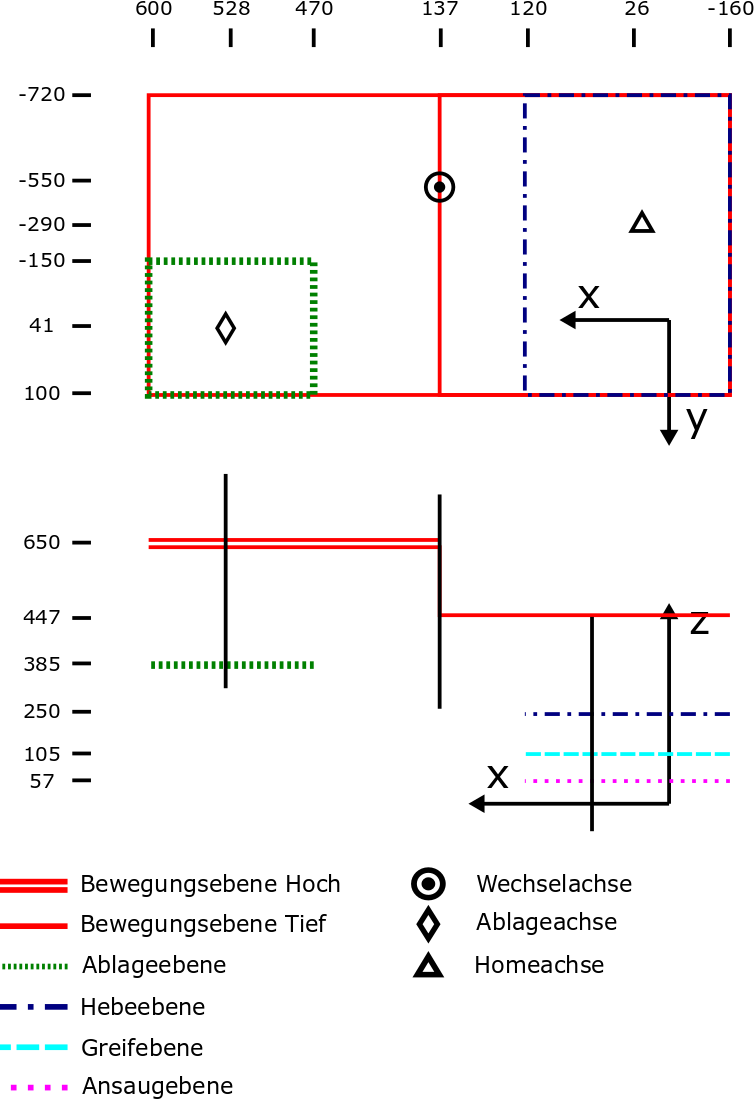


Abbildung 8-16 Ebenenbasiertes Bewegungskonzept

Tabelle 3 Ebenenbedingte Kopfrotation

|  |  |  |  |
| --- | --- | --- | --- |
| **Ebene** | **Rotation X [°]** | **Rotation Y [°]** | **Rotation Z [°]** |
| Bewegungsebene Hoch | 135 | 120 | 0 |
| Ablageebene | -15 | 167 | 0 |
| Restliche Ebenen | 180 | 0 | 0 |

Im Folgenden wird das Bewegungskonzept anhand eines Beispiels, zu sehen in Abbildung und Tabelle, verdeutlicht. Es wird der Bewegungsvorgang der beim Zuführen eines Objektes zur Messzellenrampe verwendet. Die gezeigten Befehle sind in der fertigen Implementierung in Methoden eines höheren Funktionslevels enthalten. Einzelne Befehle, wie ID 4 und ID 6 werden durch das Bewegungskonzept in mehrteilige Bewegungsabläufe, unter Berücksichtigung der Wechselachse, übersetzt um ein sicheres Erreichen der Endposition zu gewährleisten.

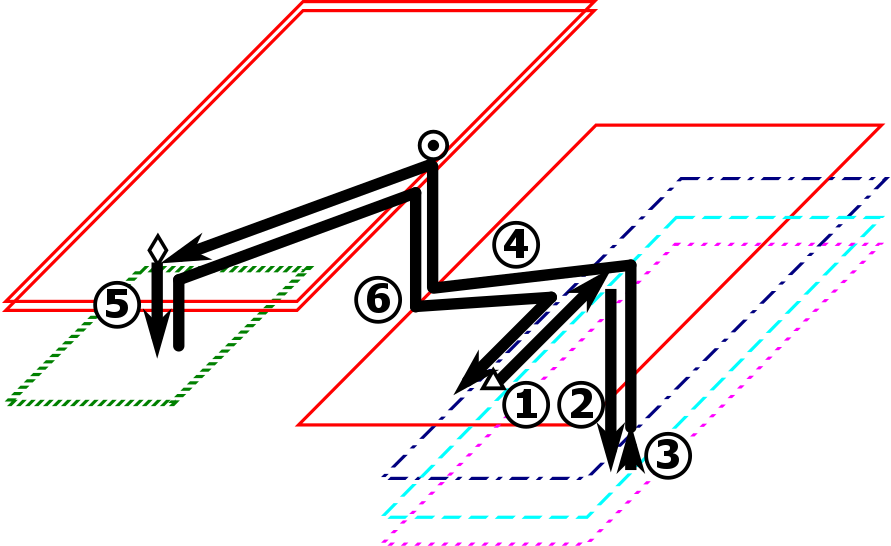


Abbildung 8-17 Beispiel Bewegungskonzept

Tabelle 4 Befehls → Aktions Zusammenhang im Beispiel Bewegungskonzept

| **ID** | **Befehl** | **Aktion** |
| --- | --- | --- |
| **1** | moveTo(X\_object,Y\_object) | *Verfährt horizontal zur Position (X\_object, Y\_object)* |
| **2** | heaveToGrippingHeight() | *Verfährt vertikal zur Greifebene* |
| **3** | heaveToLiftingHeight() | *Verfährt vertikal zur Hebeebene* |
| **4** | moveTo(X\_ramp,Y\_ramp) |  |
| **4.1** |  | *Verfährt horizontal zur Wechselachse* |
| **4.2** |  | *Verfährt vertikal entlang Wechselachse zur Bewegungsebene Hoch* |
| **4.3** |  | *Verfährt horizontal zur Ablageachse* |
| **5** | heaveToDroppingHeight() | *Verfährt vertikal entlang Ablageachse zur Ablageebene* |
| **6** | moveToHomePosition() |  |
| **6.1** |  | *Verfährt vertikal entlang Ablageachse zur Bewegungsebene Hoch* |
| **6.2** |  | *Verfährt horizontal zur Wechselachse* |
| **6.3** |  | *Verfährt vertikal entlang Wechselachse zur Bewegungsebene Tief* |
| **6.4** |  | *Verfährt horizontal zur Homeachse* |

## Förderbandkalibrierung

Für Kamerabasierte Objekterkennungs- und Klassifikationsalgorithmen ist es wichtig möglichst genau die Position des zu erkennenden, bzw. des zu klassifizierenden Objektes zu kennen. Sowohl bei der Klassifizierung in der Messzelle als auch beim Roboter, muss daher eine möglichst genaue Positionierung stattfinden. Bei beiden System findet die letzte Bewegung des Altpapierobjektes durch ein Förderband statt. Diese werden jeweils durch einen Start- und einen Stop-Befehl, welche zeitlich versetzt erfolgen, gesteuert. Um ein Objekt mit den Förderbändern genau positionieren zu können, wird eine rudimentäre Kalibrierung vorgenommen.

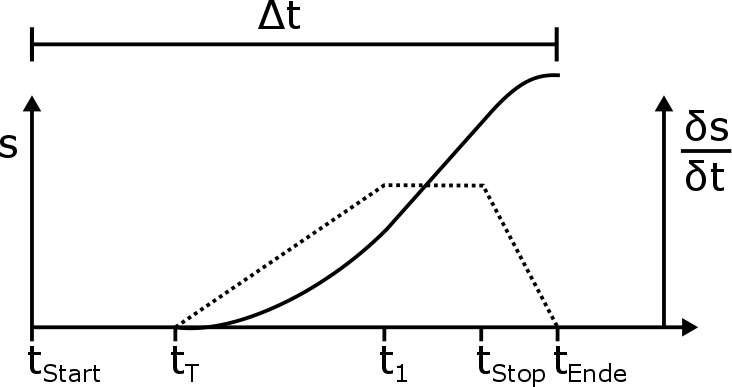


Abbildung 8-18 Angenommener Geschwindigkeits- und Positionsverlauf der Förderbänder

Es wird angenommen, dass nach dem Ausführen des Start-Befehls bei tStart eine Totzeit bis tT eintritt, bis es zur Bewegung des Förderbandes kommt. Nach einer Anlaufzeit, mit linearem Geschwindigkeitsverlauf, bis t1, hat das Förderband einen konstanten Geschwindigkeitszustand erreicht. Nach Ausführen des Stop-Befehls bei tStop wird fällt die Geschwindigkeit linear auf null. Die geförderte Distanz zum Zeitpunkt tEnde ist hiernach konstant und dient als Messgröße für die Kalibrierung.

Tabelle 5 Messdaten Förderband (Isolierung)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dauer t [s]** | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1 | 1.1 | 1.2 |
| **Distanz s [cm]** | 0 | 1 | 2 | 5.5 | 8 | 13.5 | 18 | 27.5 | 30.5 | 36.5 | 44 | 49.5 |

Tabelle 6 Messdaten Förderband (Messzelle)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dauer t [s]** | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1 | 1.1 | 1.2 |
| **Distanz s [cm]** | 0 | 1.75 | 4 | 8.75 | 14 | 20 | 26.5 | 32.5 |

Für das Förderbander wurden die Werte aus Tabelle 5 und Tabelle 6 gemessen und aus ihnen wurden mit Hilfe des Curve Fitting Tools [1] von Mathworks eine Ausgleichsrechnung durchgeführt, welche die Formel 1 und Formel 2 ergibt. Um diese Gleichungen für die Ansteuerung der Förderbänder zu verwenden ist anstatt der gemessenen Distanz die Zeit zwischen Start- und Stopp-Befehl entscheidend. Somit müssen aus Formel 1 und Formel 2 die Umkehrfunktionen in Formel 3 und Formel 4 gebildet werden. In Abbildung 8-19 und Abbildung 8-20 sind die approximierten Anfrahrts- und Förderkurven mit den, ihnen zugrunde liegenden Messdaten, abgebildet.

Formel 1 Ergebnis Ausgleichsrechnung für das Förderband Isolierung

|  |  |
| --- | --- |
|  | (1) |

Formel 2 Ergebnis Ausgleichsrechnung für das Förderband Messzelle

|  |  |
| --- | --- |
|  | (2) |

Formel 3 Umkehrfunktion von Formel 1

|  |  |
| --- | --- |
|  | (3) |

Formel 4 Umkehrfunktion von Formel 2

|  |  |
| --- | --- |
|  | (4) |

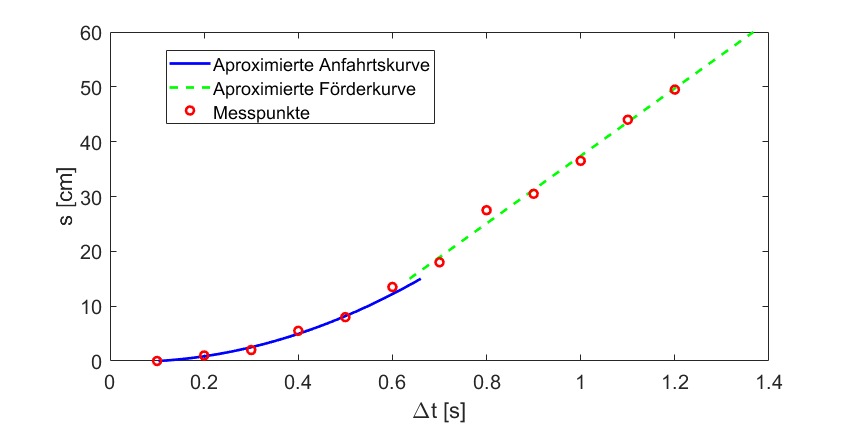


Abbildung 8-19 Approximiere Funktion für das Förderband Isolierung

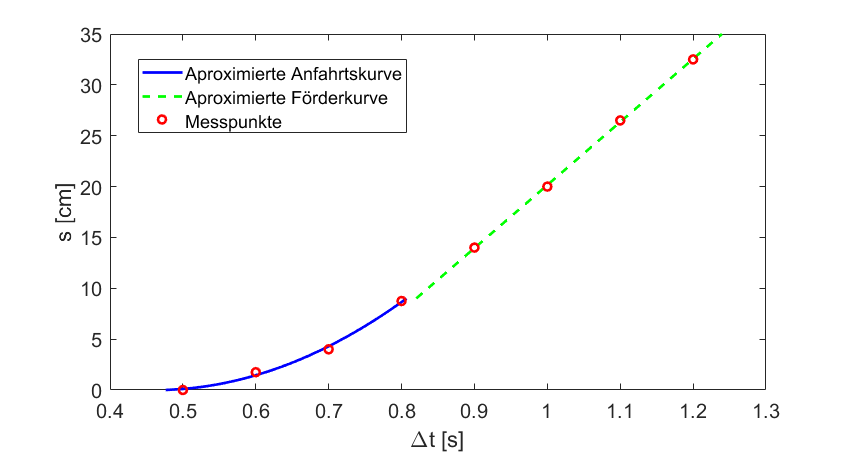


Abbildung 8-20 Approximierte Funktion für das Förderband Messzelle

# Inbetriebnahme und Validierung

* Sicherheitsmaßnahmen
* Validierung der Konstruktion

# Fazit

## Zusammenfassung

## Ausblick

# Anhang

Den Anhang auf neue Seite – ggf. neuen Abschnitt, wenn hier andere Nummerierung notwendig wird.

Nur Vorschläge, wie es aussehen könnte

Änderungen auch hier am besten über Formatvorlagen

Dieser Abschnitt ist extra nummeriert über die Abschnittsnummerierung, könnte aber auch fortlaufend wie vorherige Seiten gestaltet werden

## Klassendiagramm Neuentwicklungen

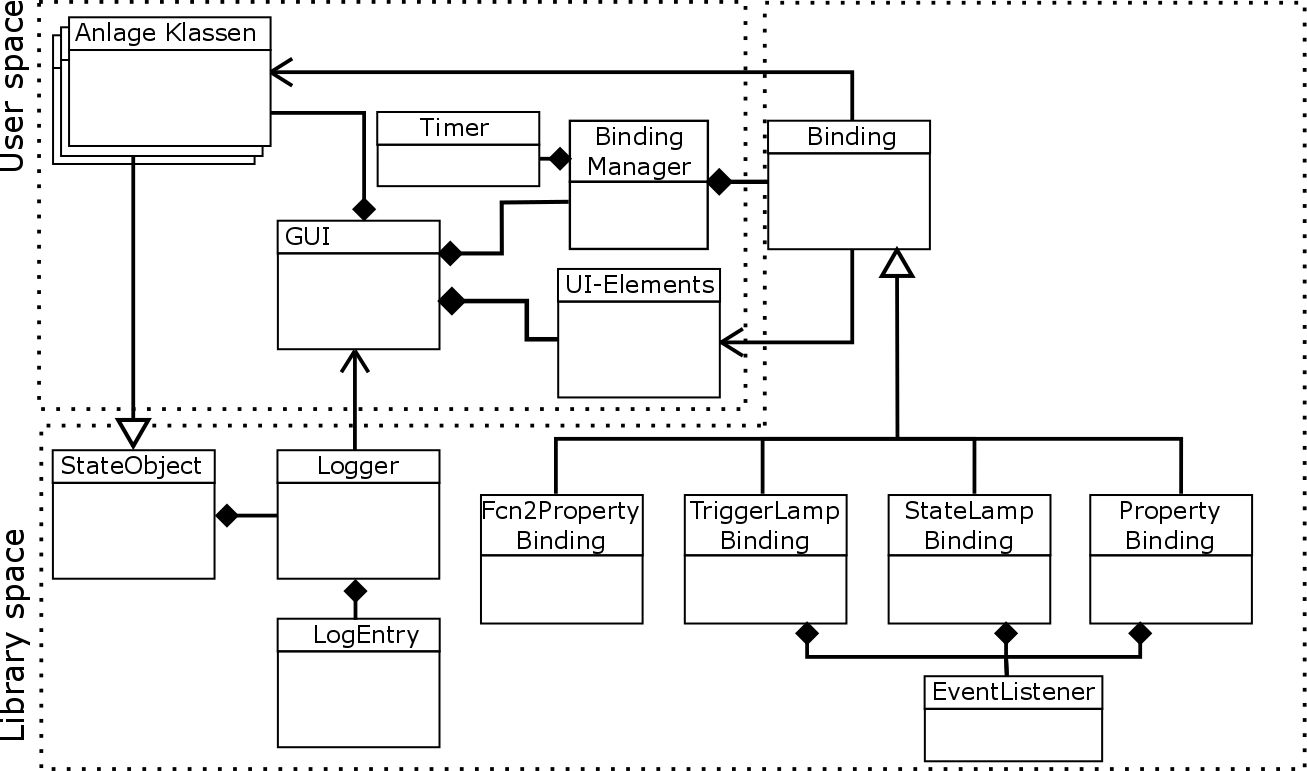
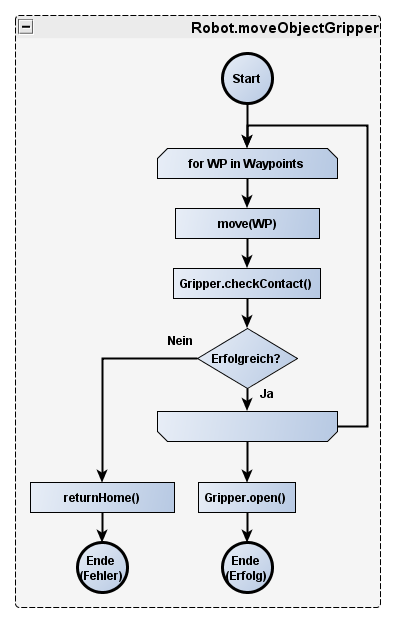
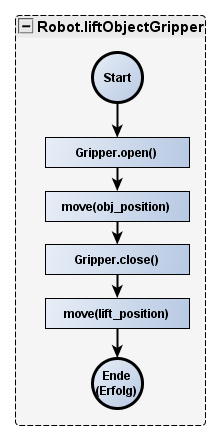
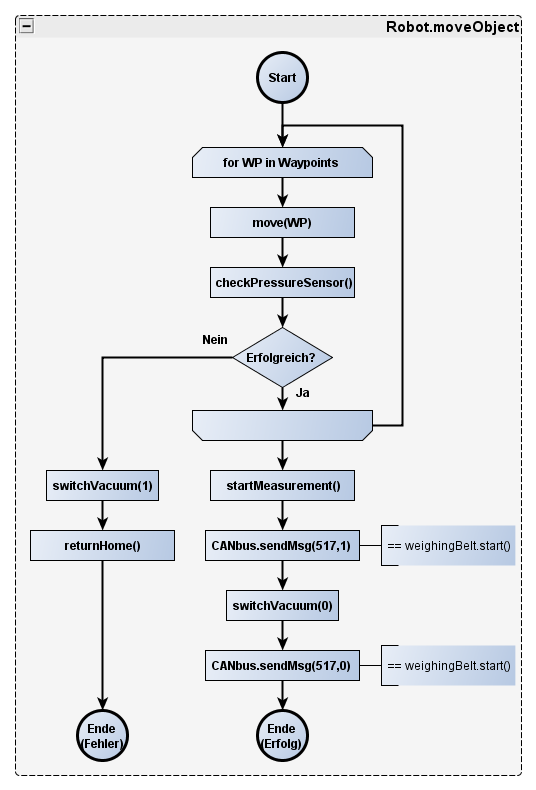
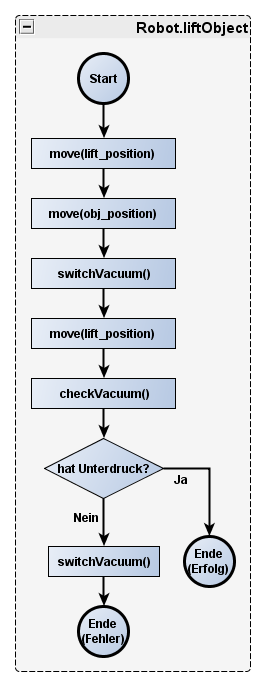
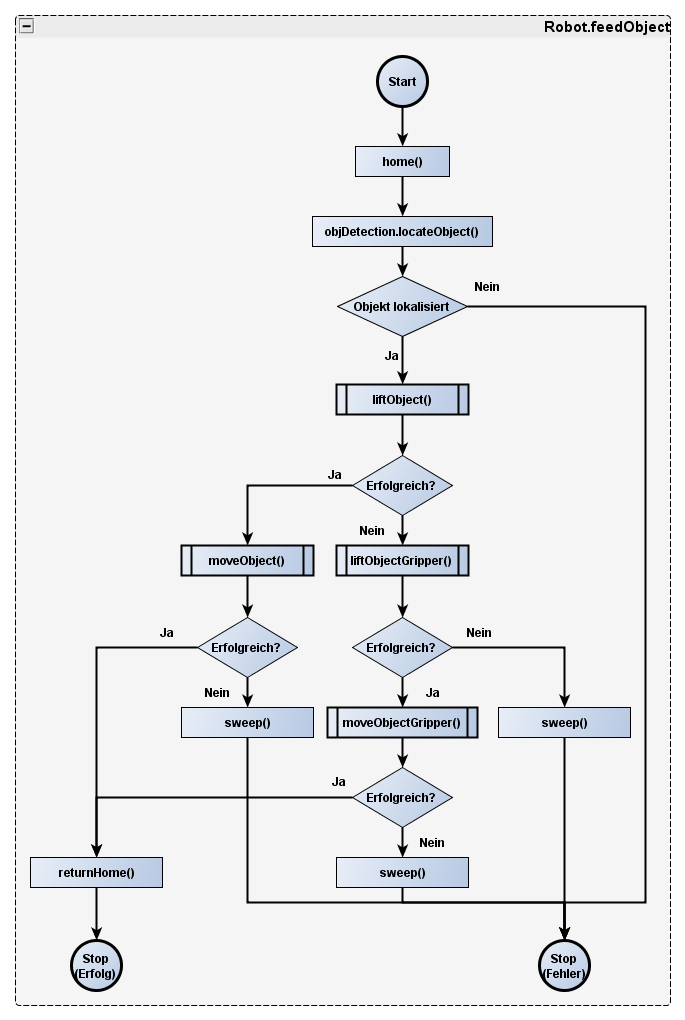
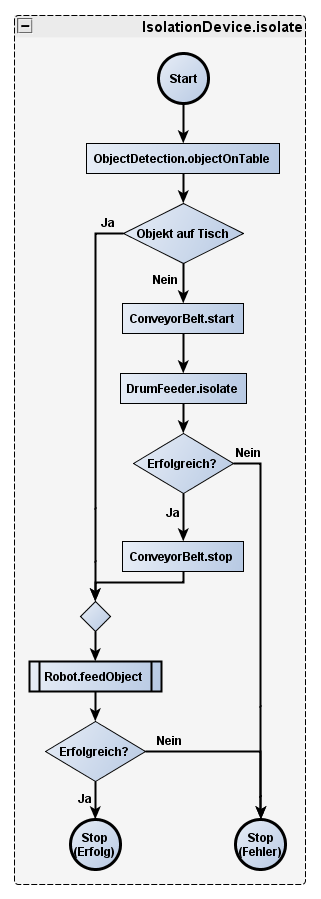
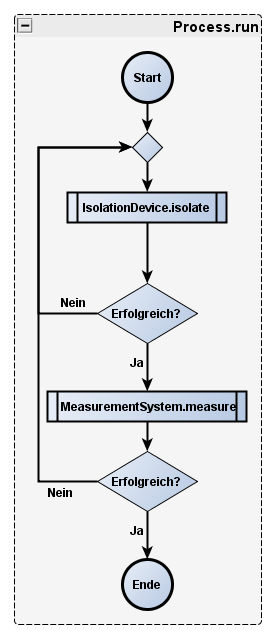
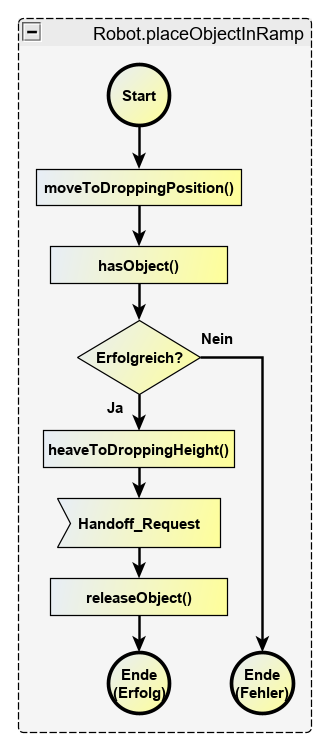
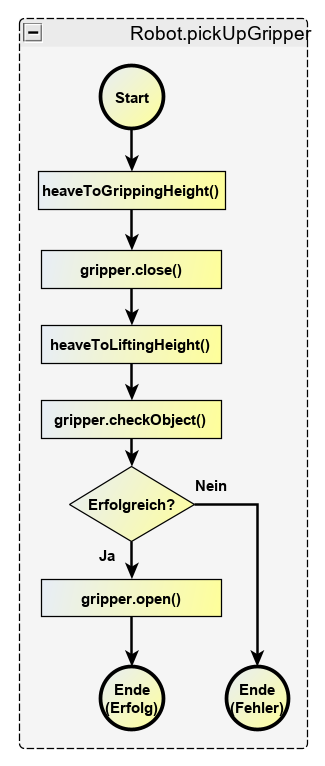
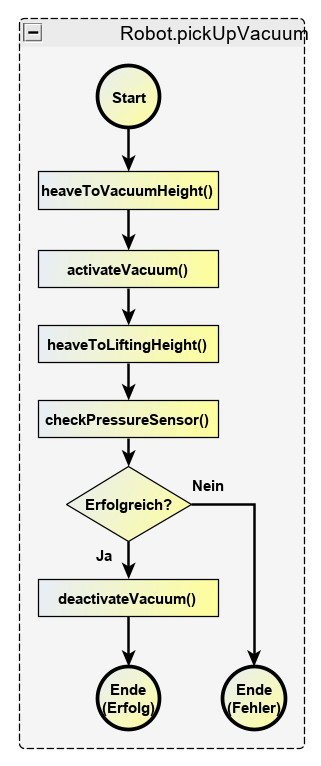
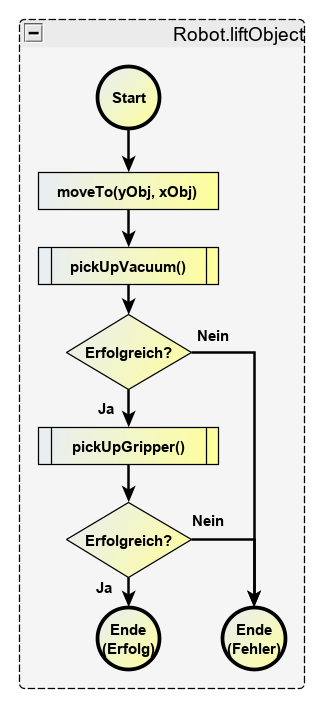
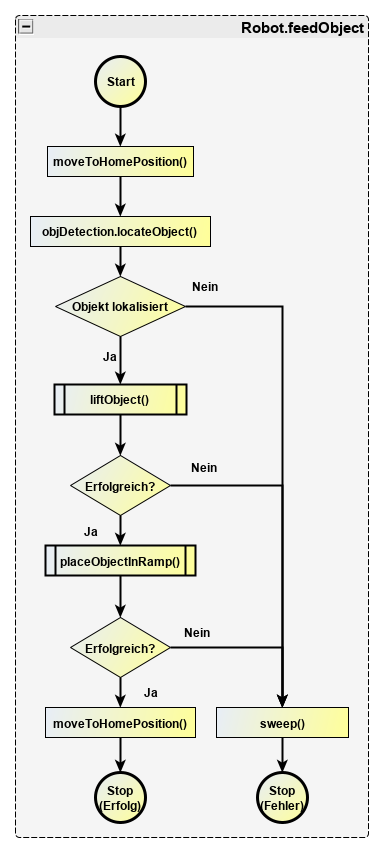


Abbildung 11-1 Klassendiagramm für Neuentwicklung

## Prozess alt



## Prozess neu



# Abbildungsverzeichnis

Abbildung 1: Bild 1; Formatierung über Formatvorlage "Beschriftung", Frontpage 9, nicht fett 3

Abbildung 2: Bild 2, Bilder mit Text in Zeile 3

# Abkürzungsverzeichnis

ff nach einer Seitenzahl: und folgende Seiten

Hrsg. Herausgeber

PISA Programme for International Student Assessment

S. Seite

u.a. unter anderem

u.s.w. und so weiter

vgl. vergleiche

z.Bsp. zum Beispiel

# Literaturverzeichnis

Baumert, Jürgen / Trautwein, Ulrich / Artelt, Cordula:   
Deutsches PISA-Konsortium (Hrsg.): PISA 2000 . Ein differenzierter Blick auf die Länder der Bundesrepublik Deutschland, Opladen 2003

Böhnisch, Lothar; Schröer, Wolfgang: Pädagogik und Arbeitsgesellschaft, Weinheim und München 2001

Boenicke, Rose / Gerstner, Hans-Peter / Tschira, Antje: Lernen und Leistung, Vom Sinn und Unsinn heutiger Schulsysteme, Darmstadt 2004

# Indexverzeichnis

Definitionen 1

Formatvorlagen 1

Schriftart 1

Text 1, 3

Überschrift 1

Das Indexverzeichnis sollte immer auf eine neue Seite, da ein neuer Abschnitt definiert werden muss wegen der 3- (oder 2-) Spalten. Wenn dieses Verzeichnis auf eine fortlaufende Seite kommt, stimmt unter Umständen die Seitennummerierung nicht mehr.

1. References

[1] MathWorks Deutschland. *Curve Fitting Tools - MATLAB & Simulink*. Retrieved April 29, 2019 from https://​de.mathworks.com​/​help/​curvefit/​interactive-and-programmatic-curve-fitting-environments.html.

1. In Matlab „Property“ genannt [↑](#footnote-ref-2)