MIXED MODE

1) Requirements – Management

2) Use Cases

3) Fallbeispiel (UC, OOA, OOD)

Dipl.-Ing. Andrea Asprian

Mixed Mode GmbH – Lochhamer Schlag 17 – 82166 Gräfelfing – www.mixed-mode.de

Inhalt und Vorgehensweise

- Einführung in das Requirements - Management
  - Requirements?
  - Requirements im Entwicklungsprozess
  - Arten von Requirements
- Requirements aufdecken mit Use Cases und UML
  - Use Cases?
  - Use Cases & UML
  - Erstellen von Use Cases
- Fallbeispiel
  - Requirements aufnehmen
  - OOA des Problems
  - Erstellen eines OO Designs

3. Fallbeispiel

zu

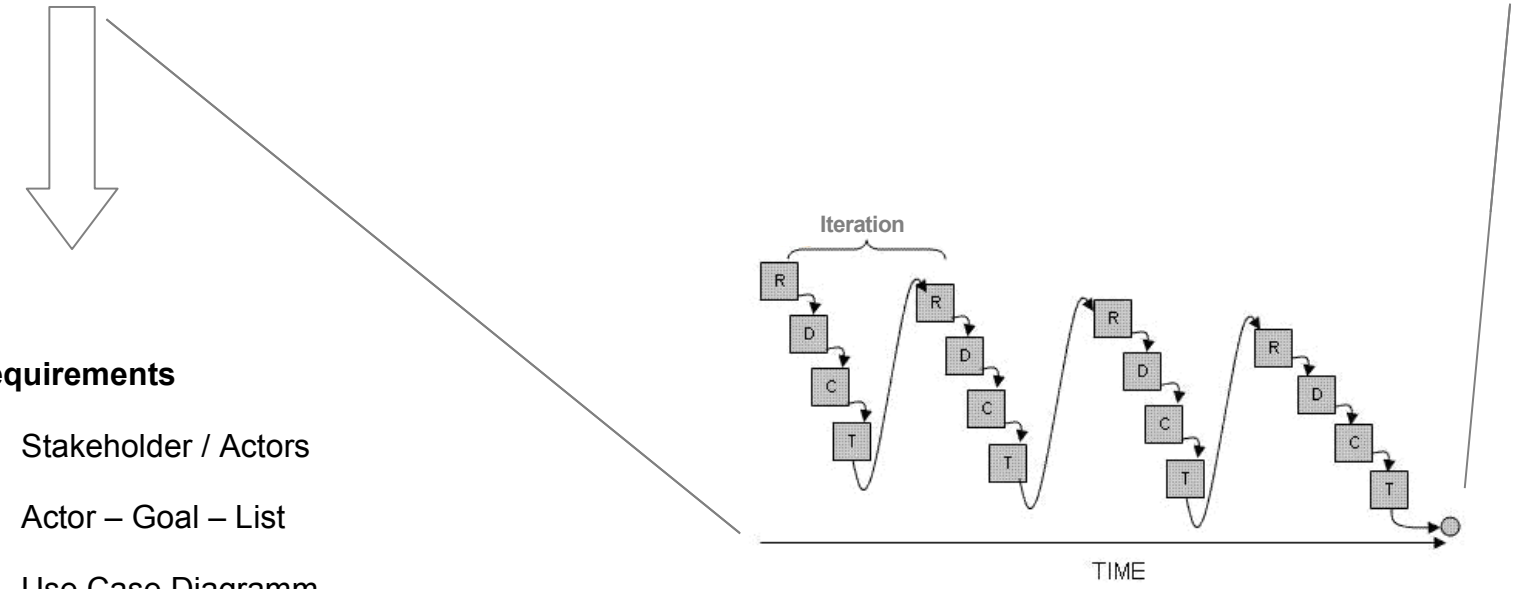
Requirements

Stakeholder / Actor	User Goals	System – Context	Requirements

OOA / OOD

OOA	OOD

Entwicklungszyklus



1) Requirements

- Stakeholder / Actors
- Actor – Goal – List
- Use Case Diagramm
- Use Case(s) (Funktionale Requirements)
- Ergänzende Spezifikation (Nichtfunktionale Requirements)

2) OOA / OOD

Beispiel: Transport – System

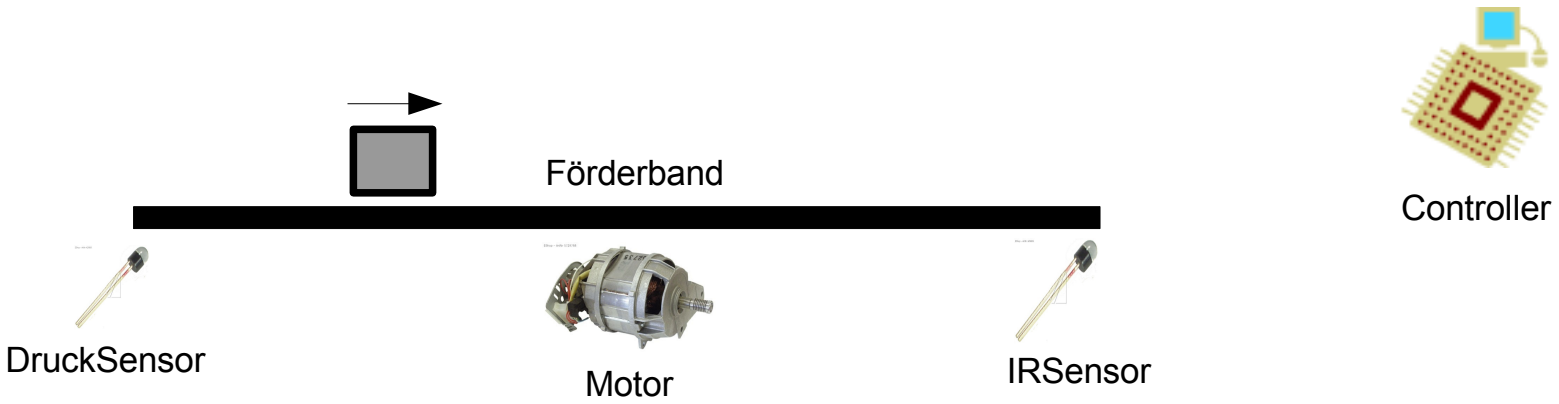
Das Förderband wird an dessen Anfang mit Transportgut beladen.

Das Gut soll bis ans Ende des Bandes transportiert werden, wo es von einem Greifer übernommen wird.

Wenn dieser nicht bereit ist, muss das Band gestoppt werden.

Ein Controller und mechanische Elemente sind bereits vorhanden.

Die Software für den Controller soll obigen Vorgang automatisieren, der bis jetzt von Hand ausgeführt wurde.



Iteration 1

Requirements

- Stakeholder / Actors
- Actor – Goal – List
- Use Case Diagramm
- Use Case(s) (Funktionale Requirements)
- Ergänzende Spezifikation (Nichtfunktionale Requirements)

Stakeholder / Actor

Wer ist am Erfolg des Projekts beteiligt?

Wer besitzt essentielle Informationen?

Wessen Interessen müssen gewahrt werden?

Stakeholder

- Auftraggeber
- Management (Finanzen)
- Projektleitung (Zeitplan)
- HW-Hersteller (Controller)
- Lieferant (mechanische Komponenten)
- Bediener
- Wartung / Technisches Personal
- Gesetzbeger (Notaus?)

Benutzer

Akteure

- Bediener
- Wartung / Technisches Personal
- DruckSensor
- IR Sensor
- Motor

User Requirements

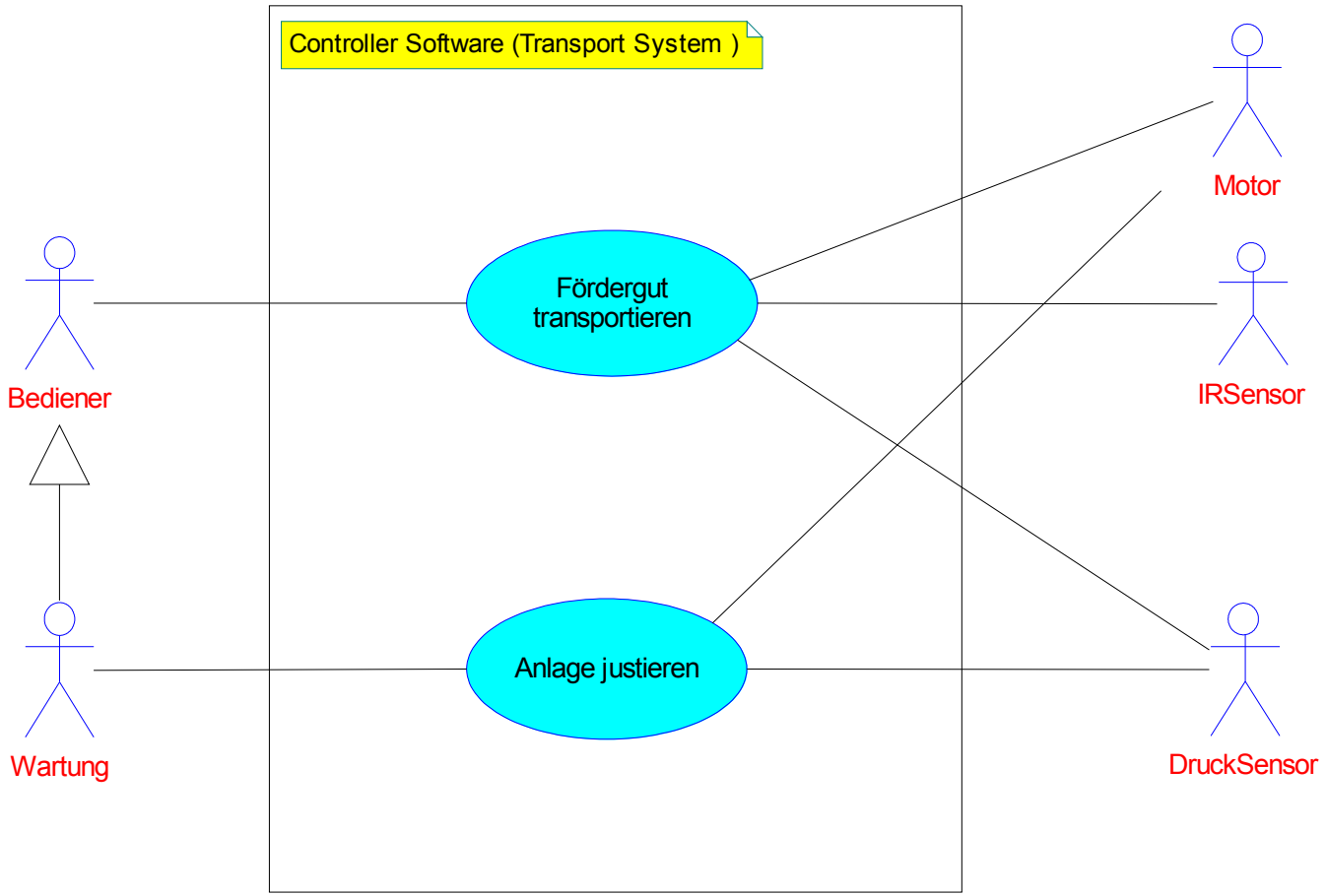
Erfüllung?

Was sind die Interessen der Benutzer?

- Fördergut transportieren
- Anlage justieren (Transportgeschwindigkeit, Mindestgewicht des Transportguts)

- aktuelle Anlagejustierung wissen
- Betriebszustand wissen
- Systemzugang beschränken

Use Case Diagramm



Use Cases

(1/2) Functional Requirements

Rang	*Use Case*	*Beschreibung*
High	**Fördergut transportieren**	Bediener belädt das Förderband. Bei ausreichender Last, setzt der Motor das Band in bewegung, bis das Gut den IR Sensor passiert. Der Motor stoppt nach maximal 50ms das Band. Das Gut wird abgehoben. Befindet sich noch ausreichend weiteres Gut auf dem Band wird der Motor wieder gestartet.
High	**Anlage justieren**	Das technische Personal stell ein: 1. die Geschwindigkeit des Förderbands 2. die Mindestlast auf dem Band damit der Motor gestartet wird.

Iteration 1

System:	Controller SW (Transport System)
Use Case:	Fördergut transportieren
Akteure:	Bediener (primär)
Haupt-Erfolgsszenario:	Transportgut passiert IRSensor, Förderband gestoppt
Trigger / Goal:	Bediener
Extension Points:	-
Include Points:	-
Nichtfunktionale Forderungen:	-
Vorbedingungen:	System ist betriebsbereit (Ruhezustand), Anlage ist justiert

Haupt-Erfolgsszenario (Basic Flow):

1. Förderband wird mit Transportgut beladen
2. DruckSensor benachrichtigt Controller
3. Controller startet Motor
4. Transportgut erreicht Förderband – Ende (IR – Sensor benachrichtigt Controller)
5. Controller stoppt Motor
6. Transportgut wird entfernt (IR – Sensor benachrichtigt Controller)
7. Förderband ist leer (Drucksensor – Wert: low)

Weiter Szenarien (Alternative Flows):

7a. Weiteres Transportgut auf Förderband (Drucksensor – Wert: high)
Einsprung bei 3

Exceptions:

2a. Drucksensor überlastet

Nachbedingungen:	System ist betriebsbereit (Ruhezustand)
Spezifische Anforderungen:	-
Offene Punkte:	Was passiert, wenn die Last auf dem Band für den Drucksensor zu groß ist?

Ergänzende Spezifikation

(2/2) Non-functional Requirements

Functionality	-
Usability	Die Anlagejustierung soll über eine graphische Benutzeroberfläche erfolgen.
Reliability	Das System darf in 4000 Betriebsstunden maximal 1 x ausfallen.
Performance	Das Stoppen des Motors erfolg max. 50 ms nach Auslösen des IR Sensors.
Supportability	Zu einem späteren Zeitpunkt muss die Portierung der Software auf ein Linux – System möglich sein.

Constraints

ID	*Constraint*
1.1	Das System muss in das bestehende Sicherheitsystem (DIN ...) integriert werden und diesem unterliegen.
1.2	Die Software wird in C/C++ programmiert.

Objektorientierte Analyse

Objektorientiertes Design

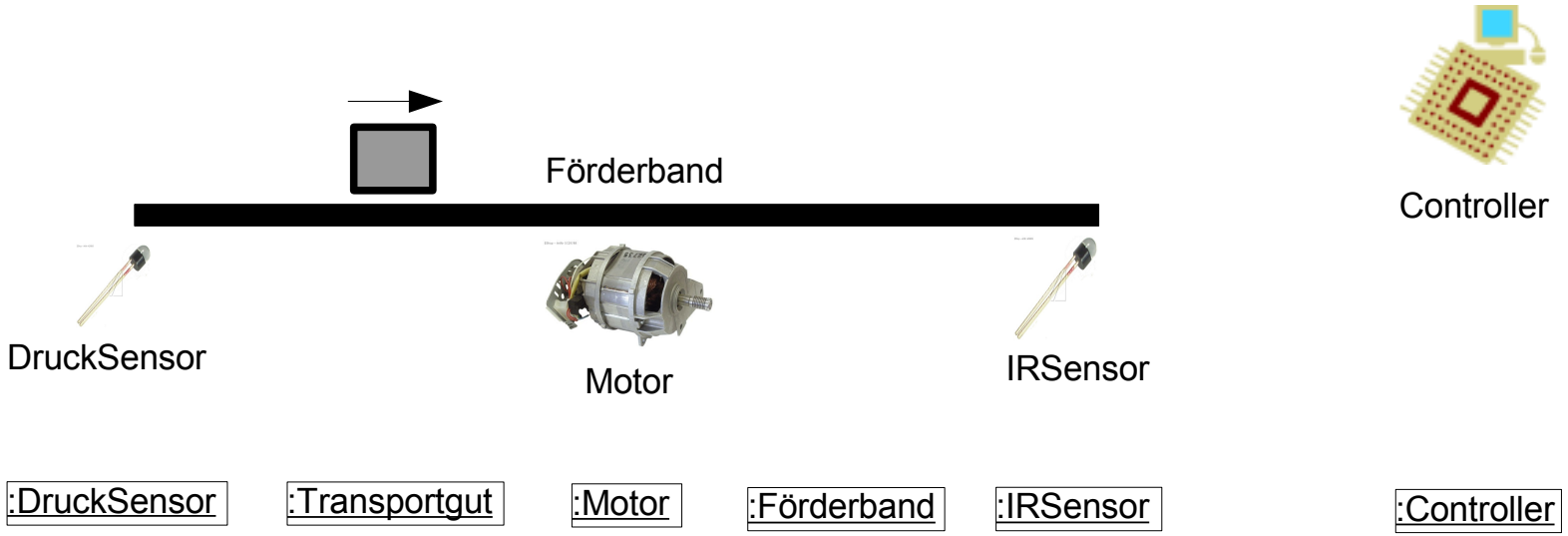
Objektorientierte Analyse

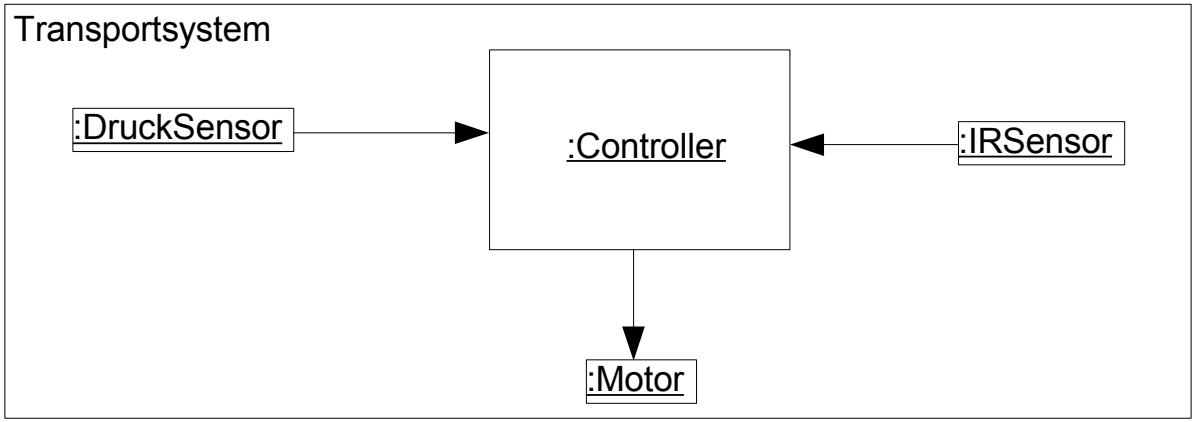
a) objektorientierte Abbildung des Systems

b) Beschreibung der Vorgänge im System

a) objektorientierte Abbildung des Systems

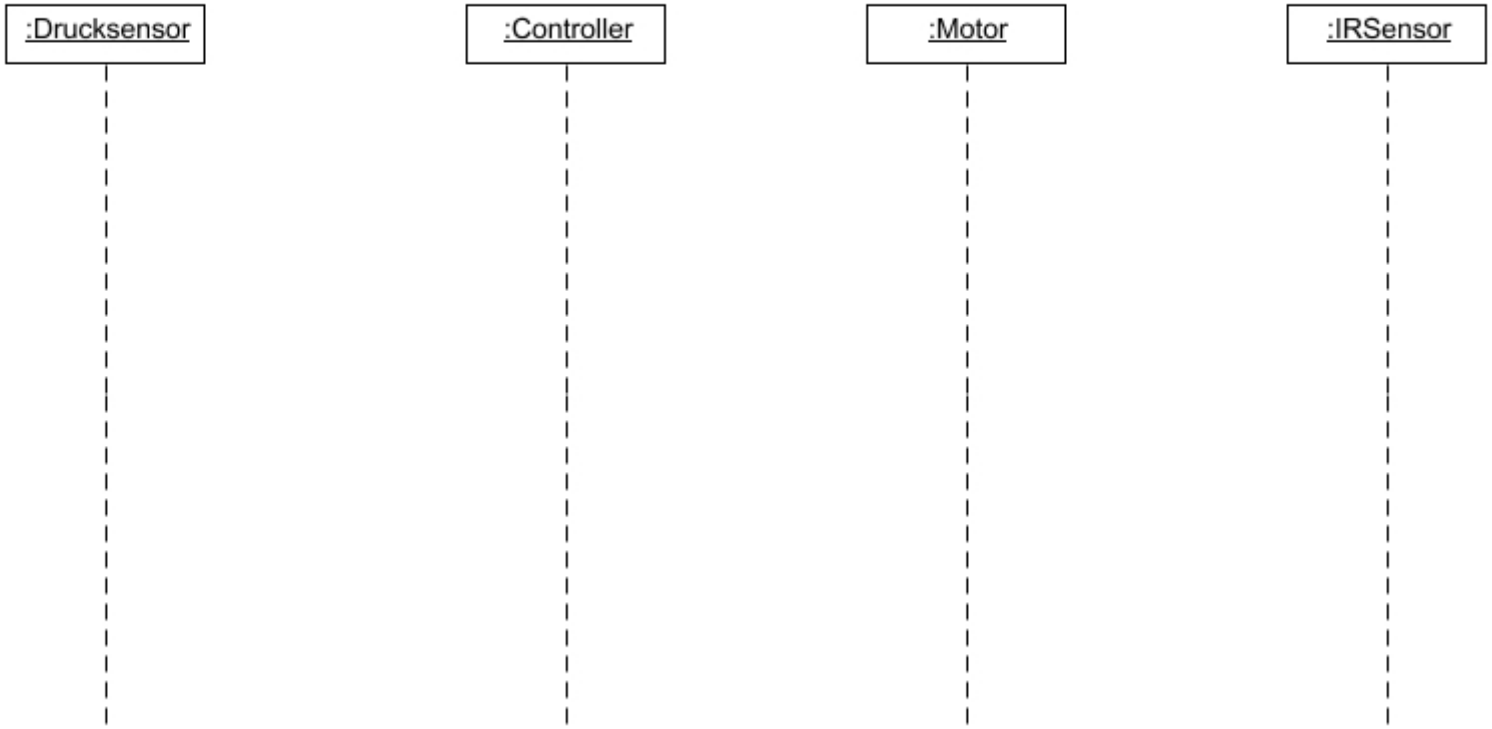
➔ graphisches Wörterbuch der Domäne





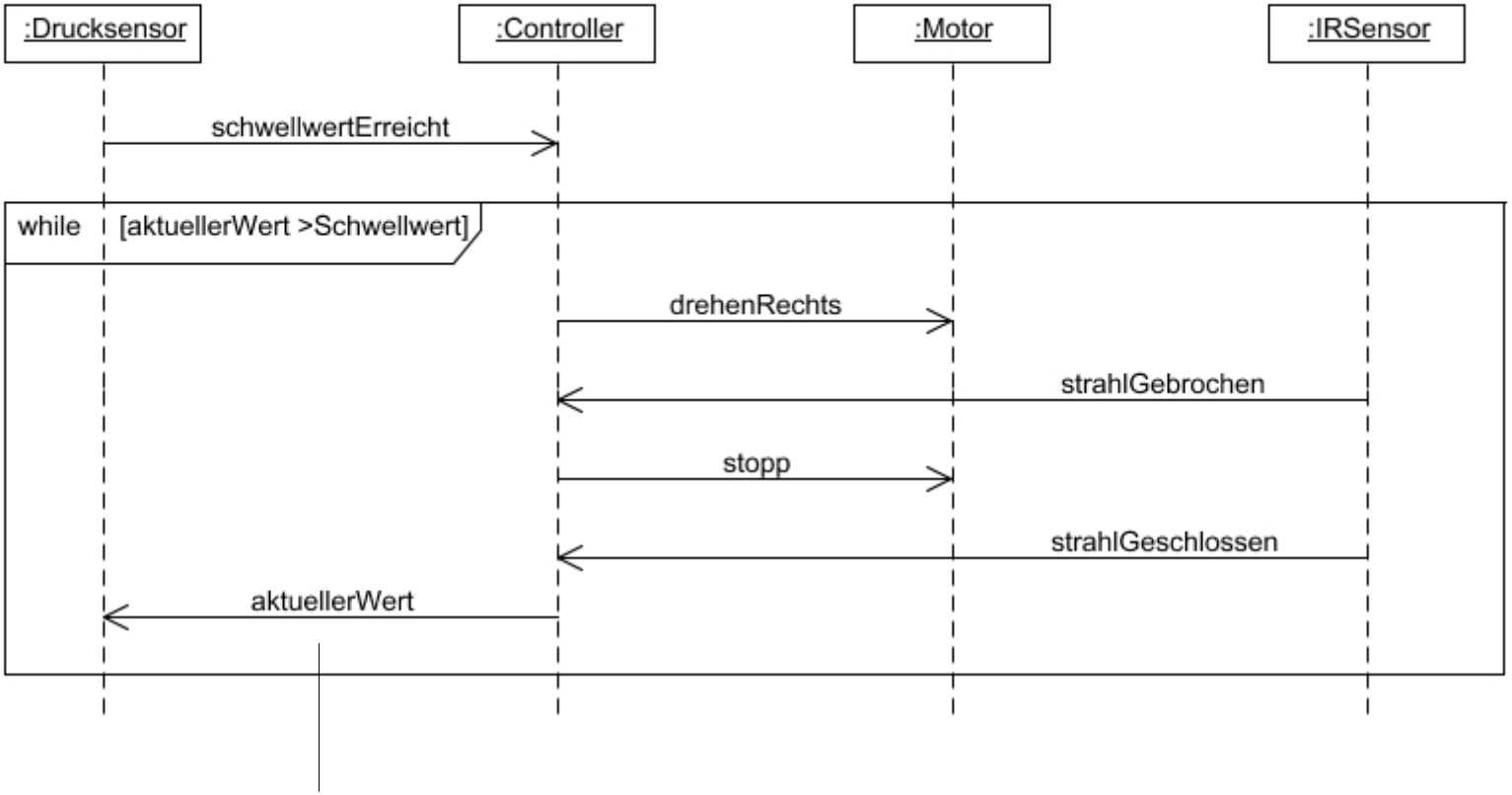
b) Beschreibung der Vorgänge im System

➔ Was tut das System?



b) Beschreibung der Vorgänge im System

➔ Was tut das System?



Nachrichten SW Methodenaufruf

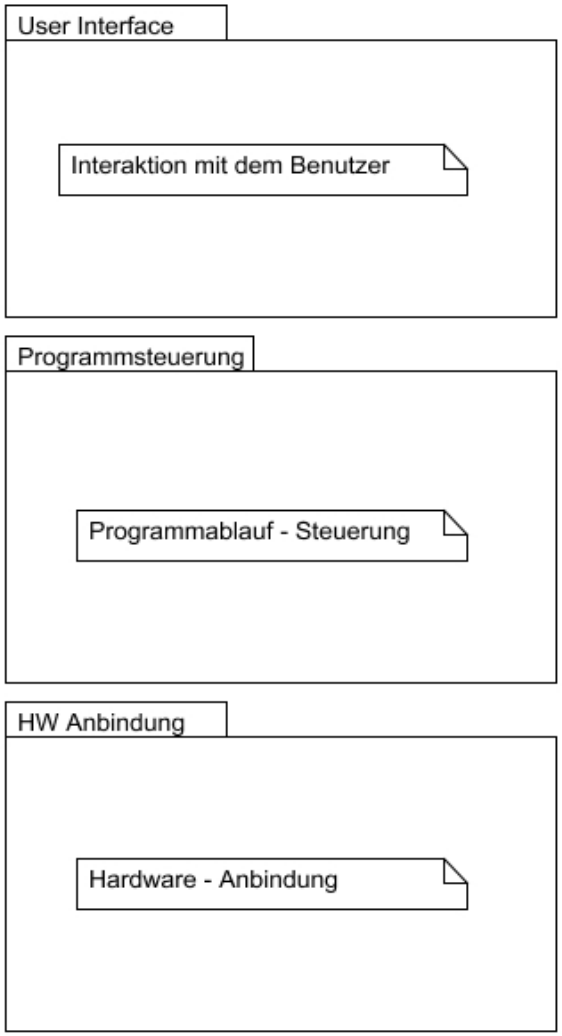
Objektorientiertes Design

a) Systemarchitektur

b) Klassendesign

a) Systemarchitektur

➔ Layer - Architektur

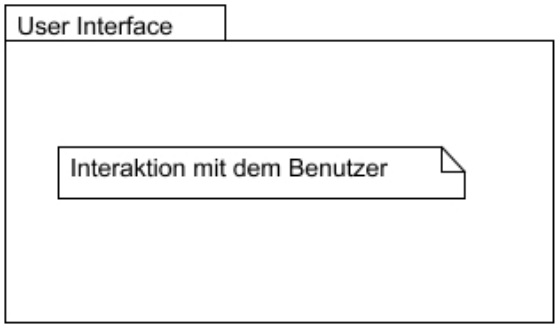


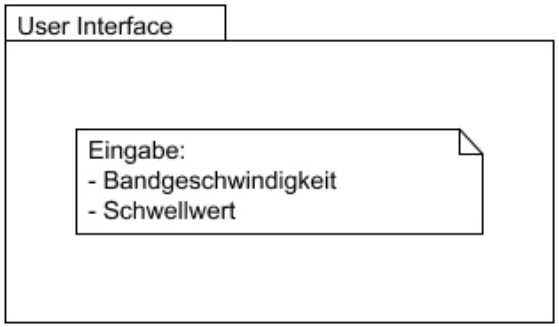
Ziele

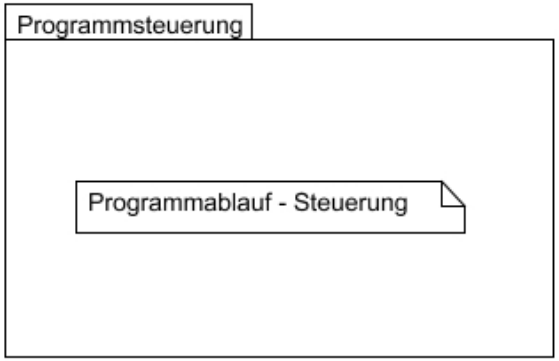
- logische Strukturierung des Gesamtsystems (Layer)
- Aufgaben der Layer
- Klassen zur Erfüllung der Aufgaben

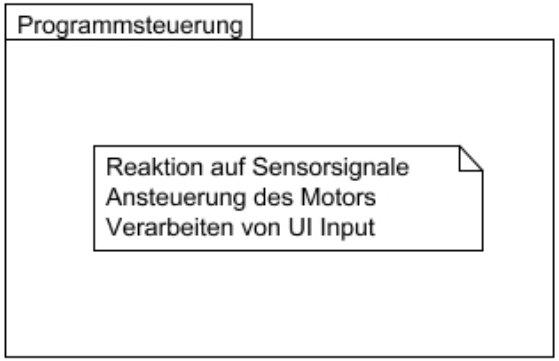
a) Systemarchitektur

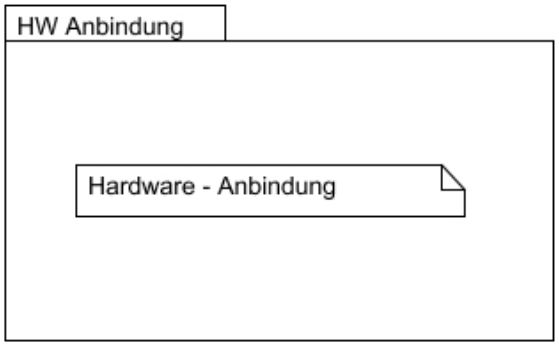
➔ Layer - Architektur

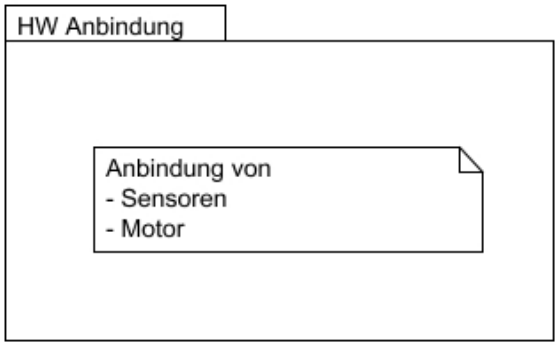












b) Klassen Design

"Gute" Klassen haben

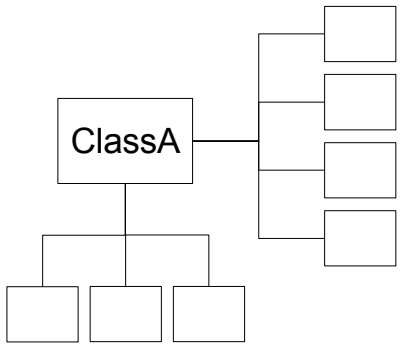
➔ größe Kohäsion

"A class should do one thing

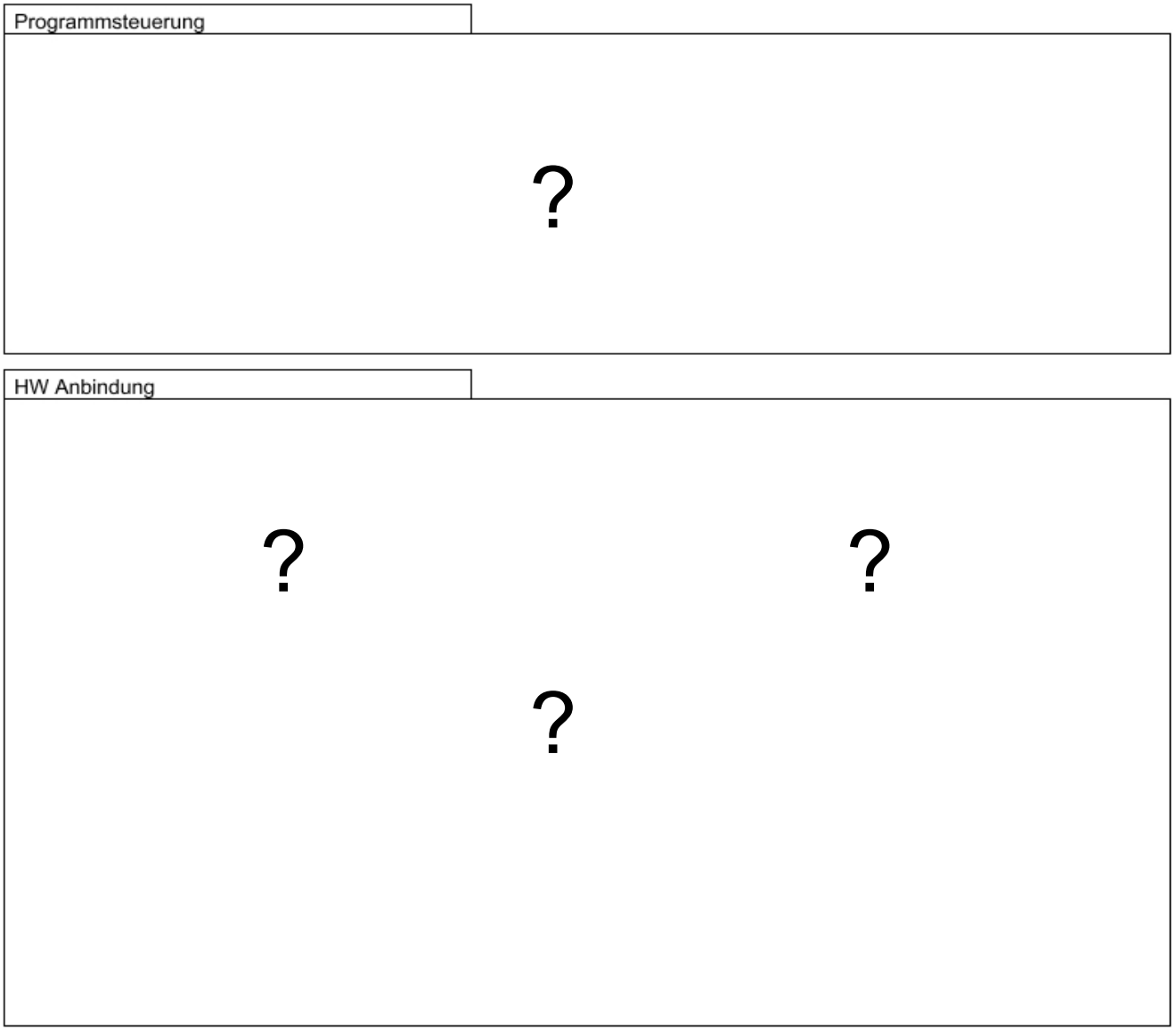
– and this should be done well!"

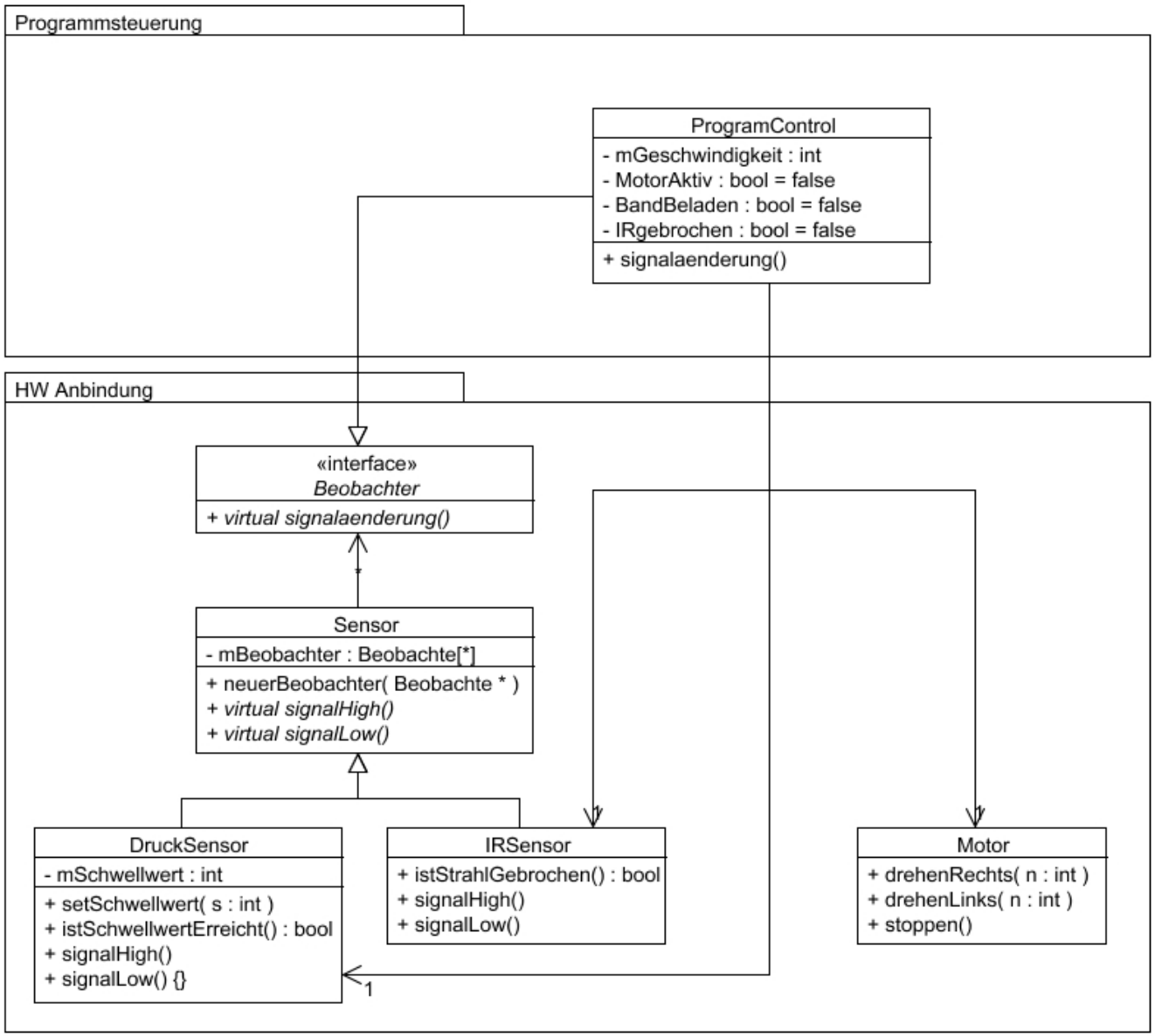
➔ geringe Abhängigkeiten

(vgl. Sensor – Steureung)



Analyse Klassen ≠ Design Klassen
1 : n





Requirements	✔
Design	✔
Implementierung	✔
Test	✔

Iteration 1 Ende

Status Quo

(Teil der) Requirements: funktionale, nicht funktionale, SystemKontext, Akteure, ...

(Teil des) Design: Systemarchitektur → erweiterbar für fongende Use Cases zB GUI

Kernklassen → risikoreiche Kernfunktionalität

Implementierung: der Kernfunktionaltität

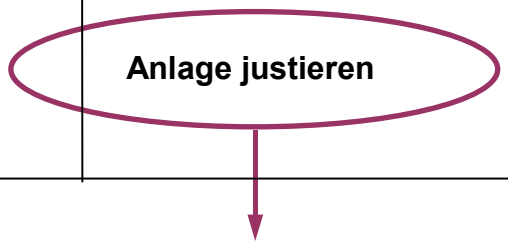
Test: risikoreicher Kern auf Realisierbarkeit getestet

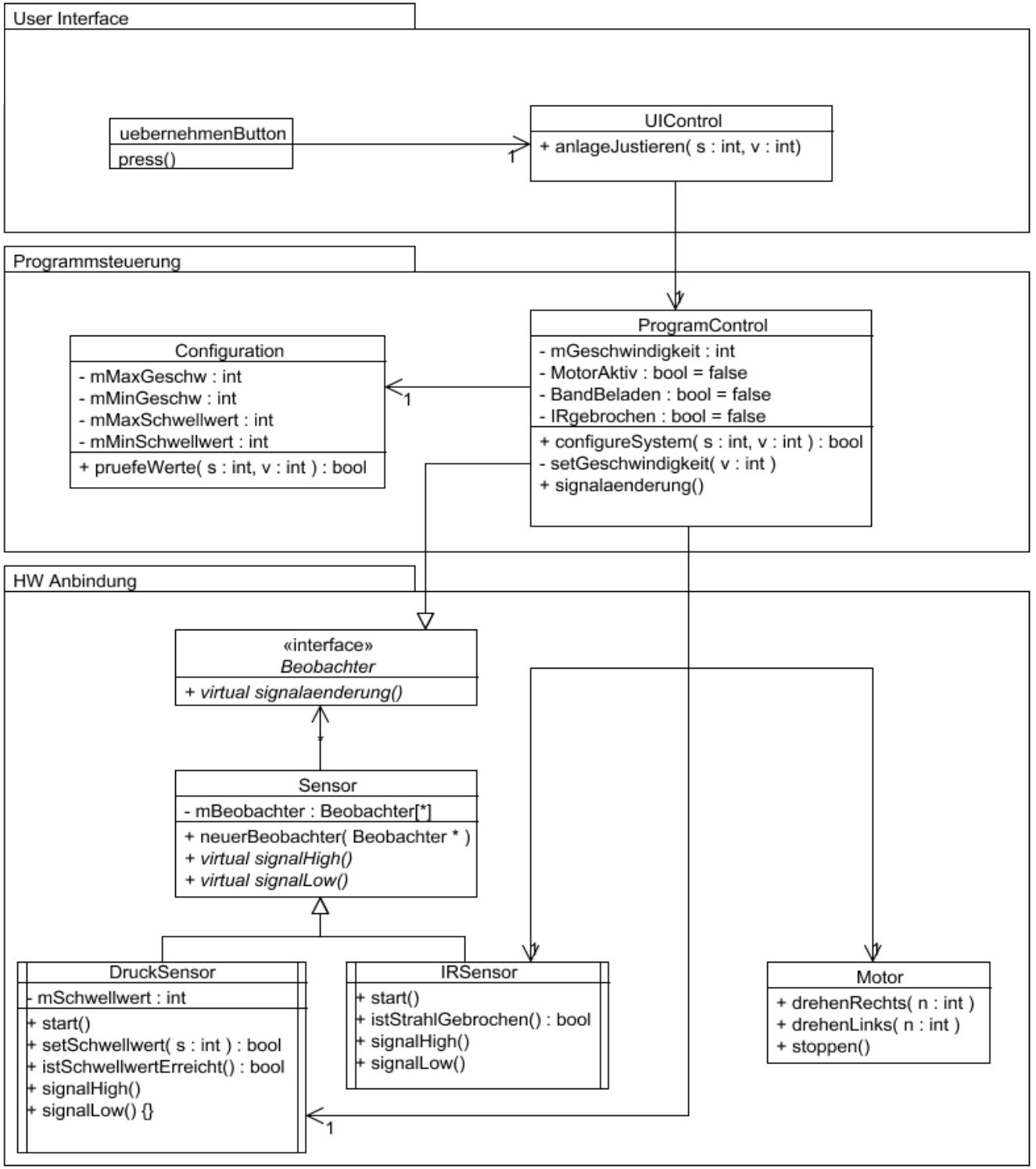
Iteration 2

Use Cases

Functional Requirements

Rang	*Use Case*	*Beschreibung*
High	**Fördergut transportieren**	Bediener belädt das Förderband. Bei ausreichender Last, setzt der Motor das Band in bewegung, bis das Gut den IR Sensor passiert. Der Motor stoppt nach maximal 50ms das Band. Das Gut wird abgehoben. Befindet sich noch ausreichend weiteres Gut auf dem Band wird der Motor wieder gestartet.
High	**Anlage justieren**	Das technische Personal stell ein: 1. die Geschwindigkeit des Förderbands 2. die Mindestlast auf dem Band damit der Motor gestartet wird.





Requirements	✔
Design	✔
Implementierung	✔
Test	✔

Iteration 2 Ende

Status Quo

(Teil der) Requirements: weiterer hochpriorer Use Case

(Teil des) Design: Systemarchitektur → erweitert durch zB GUI

weitere Klassen → zusätzliche Kernfunktionalität

Implementierung: weiterer Kernfunktionaltität

Test: Kern auf Realisierbarkeit getestet