

Dokumentation

Projektverwaltung

Änderungsübersicht

Version	Datum	Autor	Bemerkung
0.1	30.11.15	Lars Anderegg	Inhaltsverzeichnis erstellt
0.2	09.12.15	Lars Anderegg	Projektplan & Erste zwei Iterationsziele
0.3	12.12.15	Lars Anderegg	Geschäftsprozessmodellierung angefangen
0.4	15.12.15	Lars Anderegg	Anforderungsanalyse angefangen
0.5	19.12.15	Lars Anderegg	Geschäftsprozessmodellierung abgeschlossen & Neue Iterationsziele
0.6	28.12.15	Lars Anderegg	Anforderungsanalyse abgeschlossen
0.7	03.01.16	Lars Anderegg	Konfigurationsmanagement & Neue Iterationsziele
0.8	09.01.16	Lars Anderegg	Geschäftsprozessmodellierung & Anforderungsanalyse aufgrund von Review überarbeitet
0.9	10.01.16	Lars Anderegg	Testkonzept & Risikoanalyse erstellt
1.0	13.01.16	Lars Anderegg	Architektur angefangen
1.1	20.01.16	Lars Anderegg	Testfälle erarbeitet
1.2	22.01	Lars Anderegg	Testprotokoll
1.3	26.01.16	Lars Anderegg	Änderungsmanagement erstellt
1.4	02.02.16	Lars Anderegg	Architektur geändert
1.5	06.02.16	Lars Anderegg	Design-Teil angefangen
1.6	21.02.16	Lars Anderegg	Design-Teil abgeschlossen
1.7	22.02.16	Lars Anderegg	Testkonzept verbessert
1.8	24.02.16	Lars Anderegg	Einführungskonzept
2.0	26.02.16	Lars Anderegg	Finale Version erstellt

Inhaltsverzeichnis

1	Einleitung.....	3
2	Projektorganisation	3
3	Glossar	4
4	Projektplan	5
4.1	Iterationsplanung	6
5	Risikoanalyse	7
5.1	Allgemein.....	7
5.2	Analyse möglicher Probleme.....	8
6	Geschäftsprozessmodellierung	9
6.1	Geschäftsprozess.....	9
6.2	Geschäftsanwendungsfälle	10
7	Anforderungsanalyse.....	12
7.1	Systemanwendungsfälle.....	12
8	Design	20
8.1	Architektur.....	20
8.2	Model	21

8.3	Controller.....	36
8.4	View.....	38
8.5	Datenbank.....	42
8.6	Code.....	42
9	Einführung.....	43
9.1	Server.....	43
9.2	Client.....	44
9.3	Einführungsprozess.....	44
10	Qualitätsmanagement.....	45
10.1	Testkonzept.....	45
11	Konfigurationsmanagement.....	54
11.1	Hardware.....	54
11.2	Software.....	54
11.3	Daten.....	54
12	Änderungsmanagement.....	55

1 Einleitung

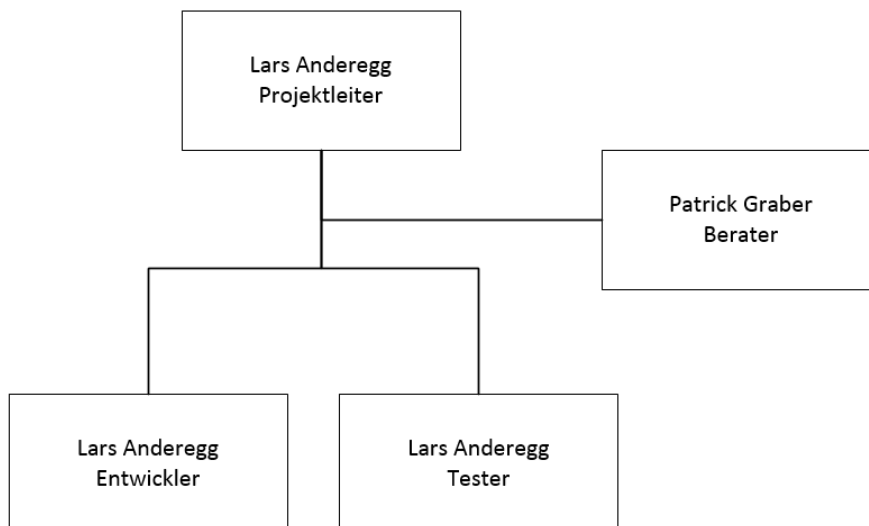
Im fünften Semester des Lehrgangs Techniker HF an der TEK0 wird eine Semesterarbeit durchgeführt. Im Rahmen der Semesterarbeit bekommt jeder Schüler einen Projektauftrag den er selbstständig umsetzen muss.

1.1 Zweck des Dokuments

Dieses Dokument beschreibt die Anforderung, Umsetzung und Controlling des Projekts.

2 Projektorganisation

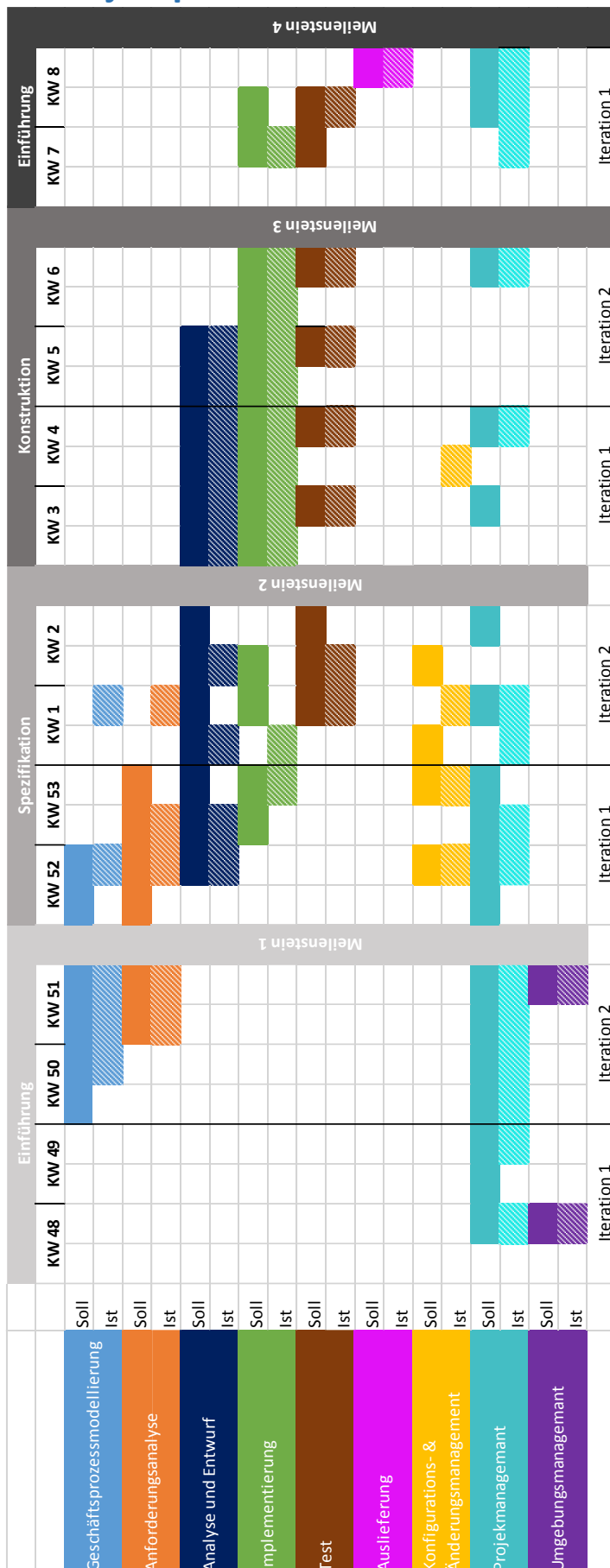
Die Projektorganisation ist wie folgt aufgebaut



3 Glossar

Begriff	Bedeutung
Dokumentenreferenz	
Ressourcen	Verfügbare Ressourcen oder verbucht Aufwände
PersonalResource	Mitarbeiterressourcen oder Aufwände
FinanceResource	Geplante und effektive Externe Kosten
Fachklasse	Klassen welche aus den Anforderungen gebildet werden und direkt abgespeichert werden.
Hilfsklasse	Hilfsklassen für die Fachklassen
MVC	Model View Controll. Architektur Prinzip zur Trennung von Daten, Ansichten und Steuerung.

4 Projektplan



4.1 Iterationsplanung

4.1.1 Einführung Iteration 1

Ziel	Erreicht	Probleme	Massnahmen
Kickoff-Meeting	OK		
Dokumentationsvorlage erstellt	OK		
Pflichtenheftvorlage erstellt	OK		
Projektplanvorlage erstellt	OK		

4.1.2 Einführung Iteration 2

Ziel	Erreicht	Probleme	Massnahmen
Pflichtenheft fertig	NOK	Zu wenig Zeit, ist aber zu 70% fertig	Noch kein kritischer Verzug, deshalb keine Massnahmen
Projektplan fertig	OK		
Geschäftsprozessmodellierung bereit für Plenum	OK		
Anforderungsanalyse bereit für Plenum	OK		

4.1.3 Spezifikation Iteration 1

Ziel	Erreicht	Probleme	Massnahmen
Geschäftsprozessmodellierung fertig	OK		
Anforderungsanalyse fertig	OK		
Git-Repository erstellen	OK		
Erster Entwurf eines Datenmodells	OK		
Auswahl des Frameworks	OK	Wurde unterschätzt	
Aufsetzen eines Workspaces	OK		

4.1.4 Spezifikation Iteration 2

Ziel	Erreicht	Probleme	Massnahmen
Datenmodell fertig	OK		
Diagramme für Use Cases	OK		
Testkonzept für Use Cases	NOK	Zu wenig Zeit	Wird in nächster Phase aufgearbeitet
Risikoanalyse erstellen	OK		
Konfigurationsmanagement ausformulieren	OK		
Mini Prototyp (Web<->Datenbank) fertig	OK		

4.1.5 Konstruktion Iteration 1

Ziel	Erreicht	Probleme	Massnahmen
UI Skizzen	OK		
Layout implementieren	NOK	Probleme mit Java-Script Frameworks	Layout mit Hilfe Bootstrap entwickeln
Saubere Schichten Trennung	NOK	Eine „falsche“ Architektur implementiert	Architektur auf die im Unterricht ändern
Änderungsmanagement erstellen	OK		
CRUD Für alle Fachklassen umsetzen	OK		
Testkonzept fertig	OK		
Erste Tests durchführen	OK		

4.1.6 Konstruktion Iteration 2

Ziel	Erreicht	Probleme	Massnahmen
Alle Funktionen fertig	OK		
Layout fertig	NOK	Layout wird nur im Firefox richtig angezeigt.	Da keine Anforderungen zu Browsern definiert ist. Ist nun Firefox der einzig unterstützte.
Alle Testfälle bestehen	OK		
Design Teil der Doku abschliessen	NOK	Zu wenig Zeit	In nächster Phase aufholen

4.1.7 Einführung Iteration 2

Ziel	Erreicht	Probleme	Massnahmen
Design Teil abschliessen	OK		
Einführungskonzept erstellen	OK		
Alle Testfälle bestehen	OK		
Dokumentation abschliessen	OK		

5 Risikoanalyse

5.1 Allgemein

5.1.1 Ziel der Risikoanalyse

Die Risikoanalyse dient zur differenzierten Betrachtung und Diskussion einzelner (oder aller) beobachteten Risiken. Sie dient vor allem für alle Projektbeteiligten als Information, welche Situationen ein Risiko darstellen könnten.

5.1.2 Projektmitarbeiter

Funktion	Name	Telefon	E-Mail
Projektleiter	Lars Anderegg	079 765 12 34	lan@software.ch

5.2 Analyse möglicher Probleme

5.2.1 Krankheit / Unfall

Id	P-01
Verantwortlich	Lars Anderegg
Beschreibung	Der Projektleiter erkrankt oder verunfallt. Er ist nicht mehr in der Lage für das Projekt zu arbeiten.
Auswirkung	Da der Projektleiter alle Aufgaben übernimmt, verzögert sich das Projekt bis er wieder in Lage ist zu arbeiten. Es kann soweit hinhalten, dass das ganze Projekt gefährdet ist.
Wahrscheinlichkeit	Gering
Tragweite	Hoch
Massnahme	Der Projektleiter versucht Gefährlich Tätigkeiten zu vermeiden und zieht sich immer warm an wenn er nach Draussen geht.

5.2.2 Datenverlust

Id	P-02
Verantwortlich	Lars Anderegg
Beschreibung	Jegliche Daten von Dokumentation bis zum Sourcecode sind auf Festplatten abgelegt. Diese können durch ein viel Zahl von Gründen nicht mehr funktionieren und auf die Daten kann nicht mehr zugegriffen werden.
Auswirkung	Die Festplatten könnten plötzlich einen Defekt haben und die Daten wären nicht mehr zu retten. Das hätte tragische Auswirkungen auf das Projekt, denn die jeweiligen Fortschritte sind in Code und Dokumentationen festgehalten.
Wahrscheinlichkeit	Gering
Tragweite	Hoch
Massnahme	Die Daten werden auf verschiedenen Servern gespeichert. Dazu wurde auf github.com ein Repository eröffnet.

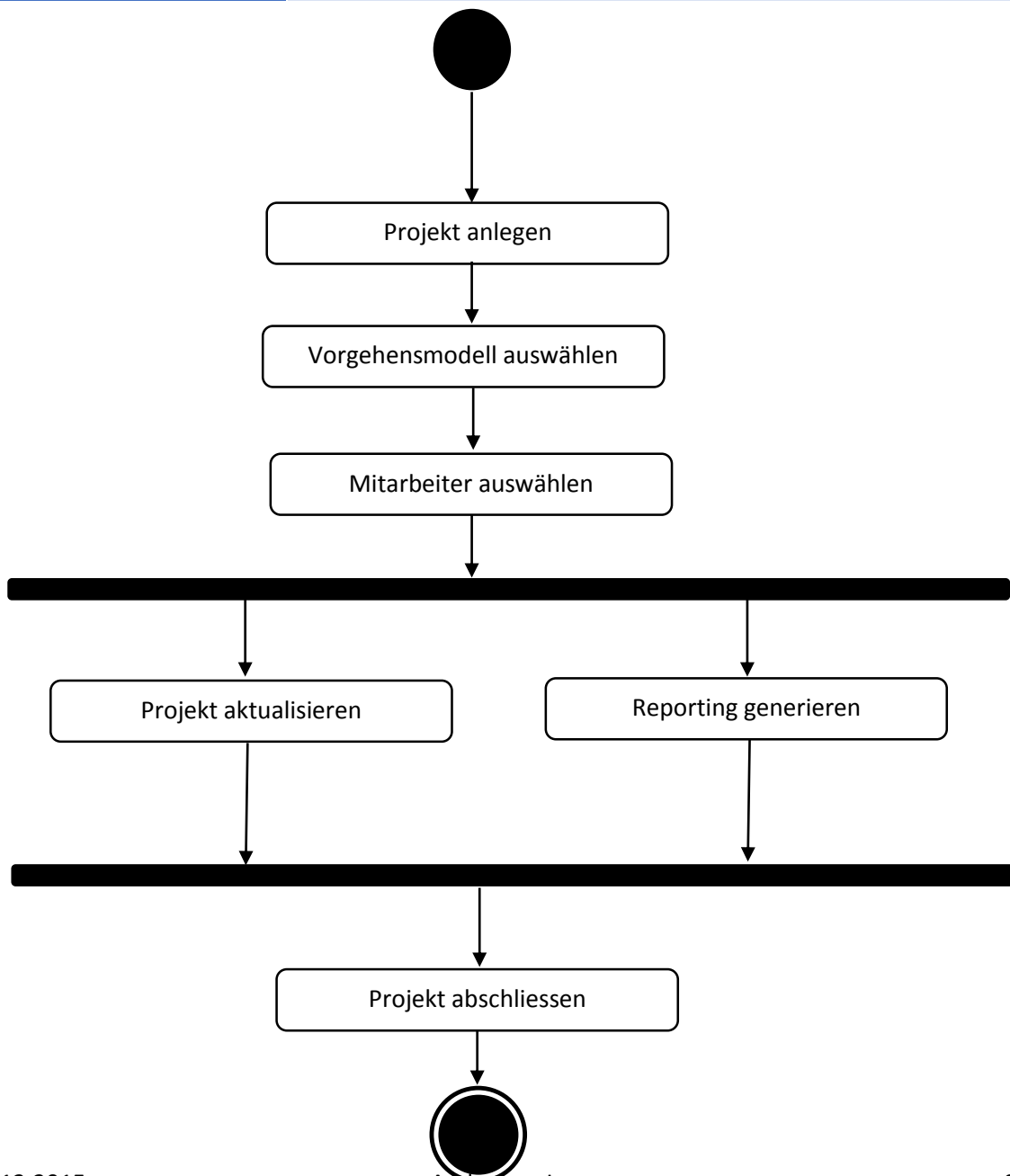
5.2.3 Zeitplanung

Id	P-03
Verantwortlich	Lars Anderegg
Beschreibung	Das Projekt hat einen fixen Abgabetermin, damit dieser erreicht werden kann wurde zu Beginn eine Planung gemacht.
Auswirkung	Wird die Planung nicht eingehalten, verschieben sich alle Tätigkeiten nach hinten. So können wichtige Funktionen oder Dokumentationsbestandteile nicht abgeschlossen werden.
Wahrscheinlichkeit	Mittel
Tragweite	Hoch
Massnahme	Durch vorausschauendes Projektmanagement und sorgfältige iterative Planung soll ein Desaster vermieden werden.

6 Geschäftsprozessmodellierung

6.1 Geschäftsprozess

Name	Projekt verwalten
Beschreibung	Alle Aktivitäten welche notwendig sind um ein Projekt verwalten zu können. Hierzu ist es notwendig das Projekt anzulegen, Vorgehensmodelle auszuwählen, und anschliessend das Projekt zu aktualisieren und Reportings zu generieren.
Enthaltene Geschäftsanwendungsfälle	<ul style="list-style-type: none"> -Projekt anlegen -Vorgehensmodell auswählen -Mitarbeiter auswählen -Projekt aktualisieren -Reporting generieren -Projekt abschliessen
Verantwortliche	Projektleiter
Beteiligte	Projektleiter, Mitarbeiter



6.2 Geschäftsanwendungsfälle

6.2.1 Projekt anlegen

Name	Projekt anlegen
Beschreibung	Projekt mit allen initialen Daten aufnehmen
Akteure	Projektleiter
Auslöser	Neues Projekt startet
Ergebnis	Projekt erstellt
Eingehende Daten	-Projektreferenz -Projektname -Projektbeschreibung -Priorität des Projekts
Vorbedingungen	
Ablauf	-Eingehende Daten erfassen
Priorität	Hoch, Projekt ist Grundlage für die folgenden Anwendungsfälle

6.2.2 Vorgehensmodell auswählen

Name	Vorgehensmodell auswählen
Beschreibung	Vorgehensmodell anlegen oder bearbeiten und schliesslich eines auswählen
Akteure	Projektleiter
Auslöser	Projekt angelegt
Ergebnis	Vorgehensmodell ausgewählt
Eingehende Daten	Daten für Vorgehensmodell
Vorbedingungen	Vorgehensmodell vorhanden
Ablauf	-Prüfen ob Vorgehensmodell vorhanden -Vorgehensmodell anlegen oder bearbeiten -Vorgehensmodell auswählen -Projektphasen generieren
Priorität	Hoch, Vorgehensmodell muss ausgewählt werden

6.2.3 Mitarbeiter auswählen

Name	Mitarbeiter auswählen
Beschreibung	Mitarbeiter in das Projekt aufnehmen.
Akteure	Projektleiter, Mitarbeiter
Auslöser	Projekt angelegt
Ergebnis	Mitarbeiter dem Projekt zugeordnet
Eingehende Daten	Funktion von Mitarbeiter
Vorbedingungen	Mitarbeiter vorhanden
Ablauf	-Mitarbeiter aussuchen -Verfügbarkeit der Mitarbeiter prüfen -Mitarbeiter zuordnen
Priorität	Hoch, ein Projekt braucht Mitarbeiter

6.2.4 Projekt aktualisieren

Name	Projekt aktualisieren
Beschreibung	Projekt mit den neusten Daten ergänzen. Hierzu gehört das verwalten der Aktivitäten, Meilensteine, Ressourcen, Dokumentreferenzen und Projektmanagement spezifische Informationen (Phasenstatus usw.).
Akteure	Projektleiter, Mitarbeiter
Auslöser	Neue Daten zum ergänzen vorhanden
Ergebnis	Projekt mit neuen Daten ergänzt
Eingehende Daten	-Aktivitäten -Meilensteine -Ressourcen -Dokumentreferenzen -Projektmanagement spezifische Informationen (Phasenstatus usw.)
Vorbedingungen	Projekt ist gestartet
Ablauf	-Aktivitäten verwalten -Meilensteine verwalten -Ressourcen verwalten -Dokumentreferenzen verwalten -Projektmanagement spezifische Informationen verwalten
Priorität	Hoch, da Kernprozess

6.2.5 Reporting generieren

Name	Reporting generieren
Beschreibung	Aus den vorhandenen Projektdaten wird ein Reporting PDF generiert. Darin ist eine Übersicht mit dem aktuellen Projektstand in Zeit und Kosten Dimension ersichtlich.
Akteure	Projektleiter
Auslöser	Projektleiter muss/will einen Report erstellen
Ergebnis	Projektübersicht mit aktuellem Stand
Eingehende Daten	Alle Daten die dem Projekt zugeordnet sind.
Vorbedingungen	Projekt ist gestartet
Ablauf	-Reporting generieren -Reporting versenden
Priorität	Niedrig, nice to have

6.2.6 Projekt abschliessen

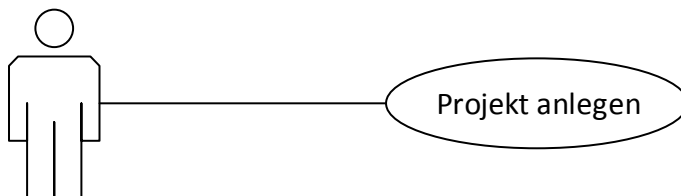
Name	Projekt abschliessen
Beschreibung	Projekt abschliessen, so dass keine Änderungen mehr möglich sind.
Akteure	Projektleiter
Auslöser	Projekt fertig
Ergebnis	Projekt kann nicht mehr geändert werden.
Eingehende Daten	
Vorbedingungen	Alle Daten für das Projekt sind erfasst
Ablauf	
Priorität	Mittel

7 Anforderungsanalyse

7.1 Systemanwendungsfälle

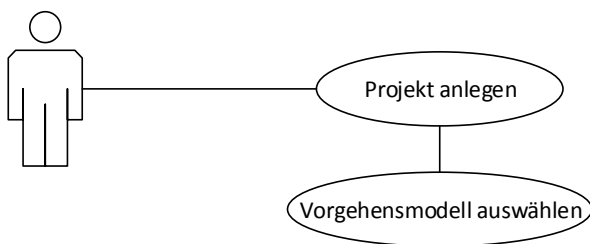
7.1.1 Projekt anlegen

Anwendungsfall	Projekt anlegen
Ziel	Erfassen aller initialen Daten
Vorbedingungen	Projektdaten vorhanden
Nachbedingungen Erfolg	Projekt ist mit erforderlichen Daten angelegt
Nachbedingungen Fehlschlag	Der Benutzer bricht das Anlegen ab
Akteur	Projektleiter
Auslöser	Neues Projekt startet
Beschreibung	Erfassen der initialen Daten
Alternativen	-
Erweiterungen	-
Priorität	hoch
Häufigkeit	Niedrig, einmal pro Projekt
Offene Punkte	-
Sonstiges	-



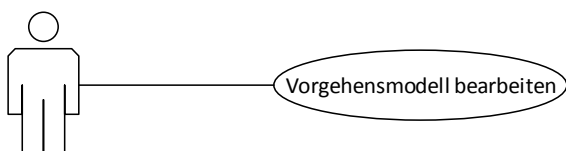
7.1.2 Vorgehensmodell auswählen

Anwendungsfall	Vorgehensmodell auswählen
Ziel	Ein Vorgehensmodell ist ausgewählt.
Vorbedingungen	Projekt vorhanden
Nachbedingungen Erfolg	Dem Projekt ist ein Vorgehensmodell zugewiesen
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter
Auslöser	Projekt angelegt
Beschreibung	Das gewünschte Vorgehensmodell wird ausgewählt. Ist das Modell noch nicht vorhanden wird ein neues angelegt oder ein bestehendes bearbeitet.
Alternativen	-Bestehendes Vorgehensmodell bearbeiten -Neues Vorgehensmodell anlegen
Erweiterungen	-
Priorität	hoch
Häufigkeit	Niedrig, einmal pro Projekt
Offene Punkte	-
Sonstiges	-



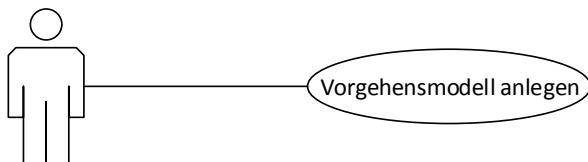
7.1.3 Bestehendes Vorgehensmodell bearbeiten

Anwendungsfall	Bestehendes Vorgehensmodell bearbeiten
Ziel	Das Vorgehensmodell ist angepasst
Vorbedingungen	Alle Projekte welche dieses Vorgehensmodell verwenden sind abgeschlossen.
Nachbedingungen Erfolg	Das Vorgehensmodell ist mit neuen Daten abgespeichert
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter
Auslöser	Kein Vorgehensmodell passt
Beschreibung	Das gewünschte Vorgehensmodell ist nicht vollständig vorhanden, deshalb wird ein bestehendes verändert.
Alternativen	-Neues Vorgehensmodell anlegen
Erweiterungen	-
Priorität	Niedrig, Workaround mit neuem Modell anlegen
Häufigkeit	Niedrig
Offene Punkte	-
Sonstiges	-



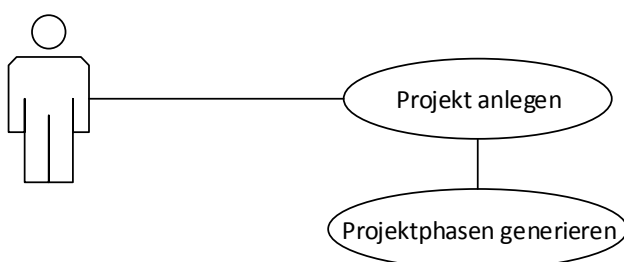
7.1.4 Neues Vorgehensmodell anlegen

Anwendungsfall	Neues Vorgehensmodell anlegen
Ziel	Neues Vorgehensmodell ist erfasst
Vorbedingungen	-
Nachbedingungen Erfolg	Das neue Vorgehensmodell ist abgespeichert
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter
Auslöser	Kein Vorgehensmodell passt
Beschreibung	Das gewünschte Vorgehensmodell ist nicht vorhanden, deshalb wird ein neues erfasst.
Alternativen	-
Erweiterungen	-
Priorität	Hoch, muss zwingend einmal gemacht werden
Häufigkeit	Niedrig
Offene Punkte	-
Sonstiges	-



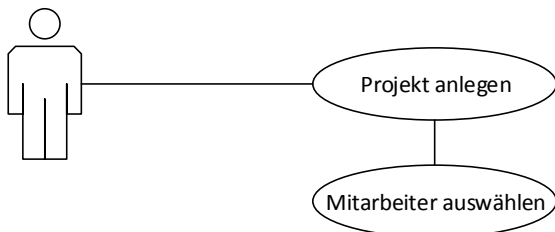
7.1.5 Projektphasen generieren

Anwendungsfall	Vorgehensmodell auswählen
Ziel	Die Projektphasen wurden erstellt und dem Projekt zugeordnet
Vorbedingungen	Projekt vorhanden und Vorgehensmodell ausgewählt
Nachbedingungen Erfolg	Die Projektphasen sind abgespeichert
Nachbedingungen Fehlschlag	Ein Systemfehler tritt auf
Akteur	System
Auslöser	Vorgehensmodell ausgewählt
Beschreibung	Aufgrund des ausgewählten Vorgehensmodells werden die Projektphasen generiert und dem Projekt zugeordnet. Für jede Phase wird ein Meilenstein erstellt welcher mit dem Phasenende verknüpft wird.
Alternativen	-
Erweiterungen	-
Priorität	Hoch, muss zwingend einmal gemacht werden
Häufigkeit	Niedrig, einmal pro Projekt
Offene Punkte	-
Sonstiges	-



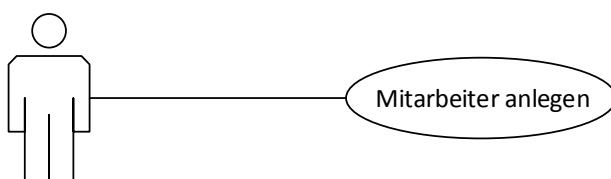
7.1.6 Mitarbeiter auswählen

Anwendungsfall	Mitarbeiter auswählen
Ziel	Mitarbeiter ist ausgewählt
Vorbedingungen	Projekt vorhanden
Nachbedingungen Erfolg	Mitarbeiter ist dem Projekt zugewiesen
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter
Auslöser	Projekt angelegt
Beschreibung	Der gewünschte Mitarbeiter wird ausgewählt. Ist dieser noch nicht vorhanden wird ein neuer angelegt oder ein bestehender bearbeitet.
Alternativen	-
Erweiterungen	-
Priorität	Hoch, muss zwingend einmal gemacht werden
Häufigkeit	Niedrig
Offene Punkte	-
Sonstiges	-



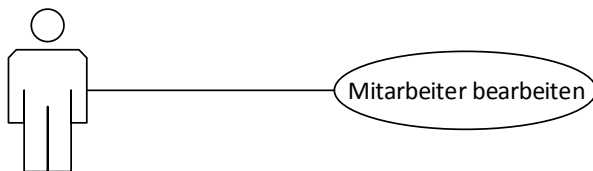
7.1.7 Mitarbeiter anlegen

Anwendungsfall	Mitarbeiter anlegen
Ziel	Neuer Mitarbeiter ist erfasst
Vorbedingungen	-
Nachbedingungen Erfolg	Der neue Mitarbeiter ist abgespeichert
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter
Auslöser	Mitarbeiter ist nicht vorhanden
Beschreibung	Der gewünschte Mitarbeiter ist nicht vorhanden, deshalb wird er erfasst.
Alternativen	-
Erweiterungen	-
Priorität	Hoch, muss zwingend einmal gemacht werden
Häufigkeit	Niedrig
Offene Punkte	-
Sonstiges	-



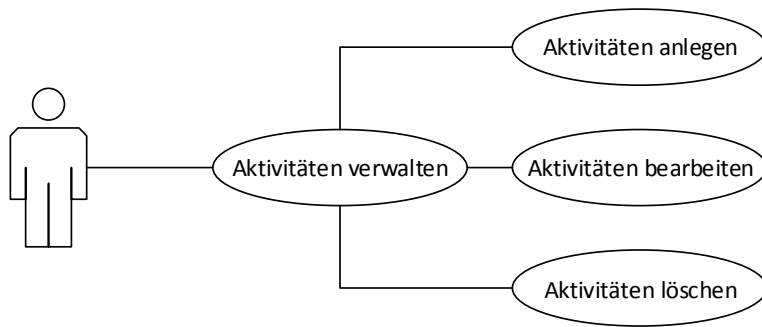
7.1.8 Mitarbeiter bearbeiten

Anwendungsfall	Mitarbeiter bearbeiten
Ziel	Mitarbeiter ist angepasst
Vorbedingungen	Mitarbeiter vorhanden
Nachbedingungen Erfolg	Der Mitarbeiter ist mit den neuen Daten abgespeichert
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter
Auslöser	Mitarbeiter ist nicht aktuell
Beschreibung	Der Mitarbeiter ist mit alten oder falschen Daten vorhanden, deshalb wird er angepasst.
Alternativen	Neuen Mitarbeiter anlegen
Erweiterungen	-
Priorität	Niedrig, Workaround mit neuem Mitarbeiter anlegen
Häufigkeit	Niedrig
Offene Punkte	-
Sonstiges	-



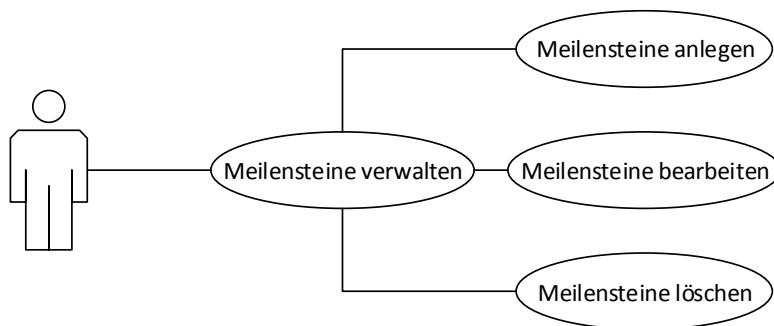
7.1.9 Aktivitäten verwalten

Anwendungsfall	Aktivitäten verwalten
Ziel	Aktivitäten sind abgespeichert
Vorbedingungen	Projektphase vorhanden
Nachbedingungen Erfolg	Die Aktivitäten sind mit den aktuellen Daten abgespeichert.
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter oder Mitarbeiter
Auslöser	Neue Aktivität muss erfasst werden oder bestehende muss bearbeitet / gelöscht werden.
Beschreibung	Eine neue Aktivität wird erfasst oder eine bestehende bearbeitet / gelöscht.
Alternativen	-
Erweiterungen	-
Priorität	Hoch, Kernprozess
Häufigkeit	Hoch
Offene Punkte	-
Sonstiges	-



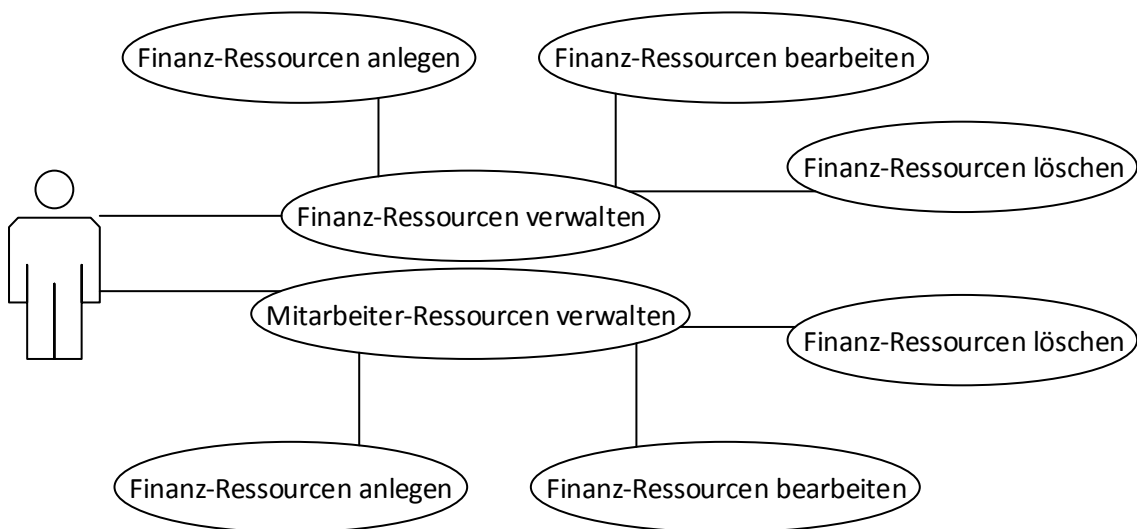
7.1.10 Meilensteine verwalten

Anwendungsfall	Meilensteine verwalten
Ziel	Meilensteine sind abgespeichert
Vorbedingungen	Projektphase vorhanden
Nachbedingungen Erfolg	Die Meilensteine sind mit den neuen Daten abgespeichert
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter
Auslöser	Neuer Meilenstein muss erfasst werden oder bestehender muss bearbeitet / gelöscht werden.
Beschreibung	Für eine Projektphase wird ein neuer Meilenstein wird erfasst oder eine bestehender bearbeitet / gelöscht. Fixe Meilenstein können nicht bearbeitet / gelöscht werden.
Alternativen	-
Erweiterungen	-
Priorität	Mittel
Häufigkeit	Niedrig
Offene Punkte	-
Sonstiges	-



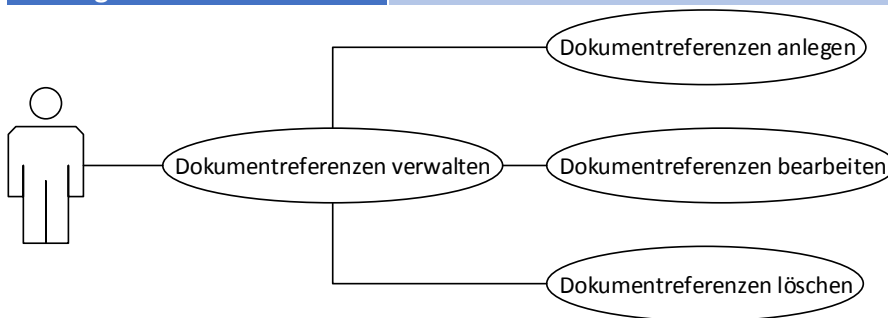
7.1.11 Ressourcen verwalten

Anwendungsfall	Ressourcen verwalten
Ziel	Ressourcen sind abgespeichert
Vorbedingungen	Aktivitäten sind vorhanden
Nachbedingungen Erfolg	Die Ressourcen sind mit den neuen Daten abgespeichert
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter oder Mitarbeiter
Auslöser	Neue Ressource muss erfasst werden oder bestehende muss bearbeitet / gelöscht werden.
Beschreibung	Für eine Aktivität wird eine Ressource bearbeitet / gelöscht. Die Ressource beinhaltet entweder Mitarbeiteraufwände oder externe Kosten.
Alternativen	-
Erweiterungen	-
Priorität	Hoch, Kernprozess
Häufigkeit	Hoch
Offene Punkte	-
Sonstiges	-



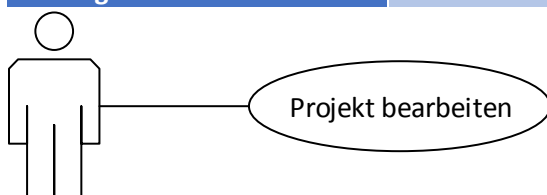
7.1.12 Dokumentreferenzen verwalten

Anwendungsfall	Dokumentreferenzen verwalten
Ziel	Dokumentreferenzen sind abgespeichert
Vorbedingungen	Projekt, Phase oder Aktivität vorhanden
Nachbedingungen Erfolg	Die Dokumentreferenzen sind mit den neuen Daten abgespeichert
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter oder Mitarbeiter
Auslöser	Dokumentreferenz muss erfasst oder bearbeitet / gelöscht werden
Beschreibung	Für ein Projekt, Phase oder eine Aktivität wird eine Dokumentreferenz erfasst, bearbeitet oder gelöscht.
Alternativen	-
Erweiterungen	-
Priorität	Mittel
Häufigkeit	Mittel
Offene Punkte	-
Sonstiges	-



7.1.13 Projektmanagement spezifische Informationen verwalten

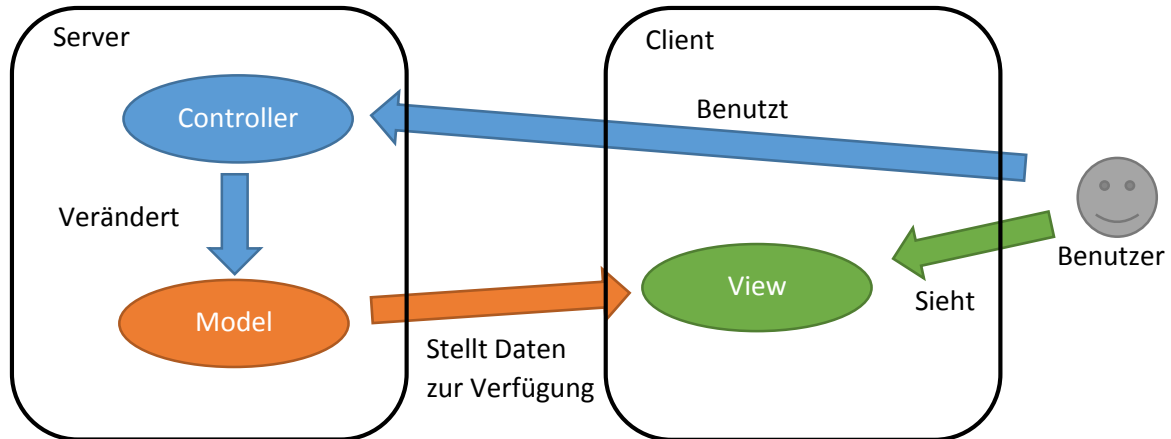
Anwendungsfall	Projektmanagement spezifische Informationen verwalten
Ziel	Projekt mit aktuellen Daten abgespeichert
Vorbedingungen	Projekt vorhanden
Nachbedingungen Erfolg	Das Projekt ist mit den neuen Daten abgespeichert
Nachbedingungen Fehlschlag	Der Benutzer bricht den Vorgang ab
Akteur	Projektleiter
Auslöser	Projekt ist nicht mehr aktuell
Beschreibung	Das Projekt wird mit den neusten Daten aktualisiert.
Alternativen	-
Erweiterungen	-
Priorität	Mittel
Häufigkeit	Niedrig
Offene Punkte	-
Sonstiges	-



8 Design

8.1 Architektur

Die Architektur richtet sich stark nach den Model View Controller Prinzip (MVC).



Die Applikation wird als eine Web-Applikation entwickelt, was letztendlich bedeutet, dass die Schichten Model & Controller in einem Web-Server laufen und die View im Browser angezeigt wird. Als Kommunikation dazwischen wird http eingesetzt. Weitere Details zu den drei Schichten sind in den folgenden Kapiteln erläutert. Diese Architektur wurde gewählt, weil sie weit verbreitet ist und auf der Client-Seite einen hohen Grad an Plattformunabhängigkeit bietet. Ändern sich die Anforderungen an Funktionen und Daten, kann dies an zentraler Stelle im Server angepasst werden. So muss kein Benutzer jemals etwas installieren und arbeitet immer mit der aktuellsten Version.

8.1.1 Eingesetzte Frameworks und Plugins

Das Rad soll nicht neu erfunden werden. Deshalb werden für standard Funktionen Plugins und Frameworks von Drittanbietern eingesetzt. Folgende Tabelle zeigt die wichtigsten

Plugin	Anbieter	Zweck
Spring	http://spring.io/	Framework welches Allgemein die Programmierung vereinfacht.
Spring MVC	http://spring.io/	Framework für die Erstellung von Webanwendungen.
Spring Roo	http://spring.io/	Hilft beim Erstellen einer Spring-Applikation. Generiert z.B. Standard-Konfigurationen für eine Webanwendung.
Hibernate	RedHat (http://hibernate.org/)	Übernimmt das Mapping von Java-Objekten auf eine relationale Datenbank.
JUnit	http://junit.org/	Framework zum Testen, besonders für Unit-Tests sehr hilfreich.
Log4J	http://logging.apache.org/	Framework zum Loggen

8.2 Model

Das Model hat die Verantwortlichkeit der Daten und der Geschäftslogik. Es besteht im Wesentlichen aus zwei Arten von Klassen.

- Fachklassen
 - ProcessModel
 - Employee
 - Project
 - Phase
 - Milestone
 - Activity
 - Resource
 - PersonalResource
 - FinanceResource
 - DocumentReference
- Hilfsklassen
 - ISummedResources
 - ITimeBoxed
 - PhaseChild
 - ResourceCollector
 - TimeBoxedData

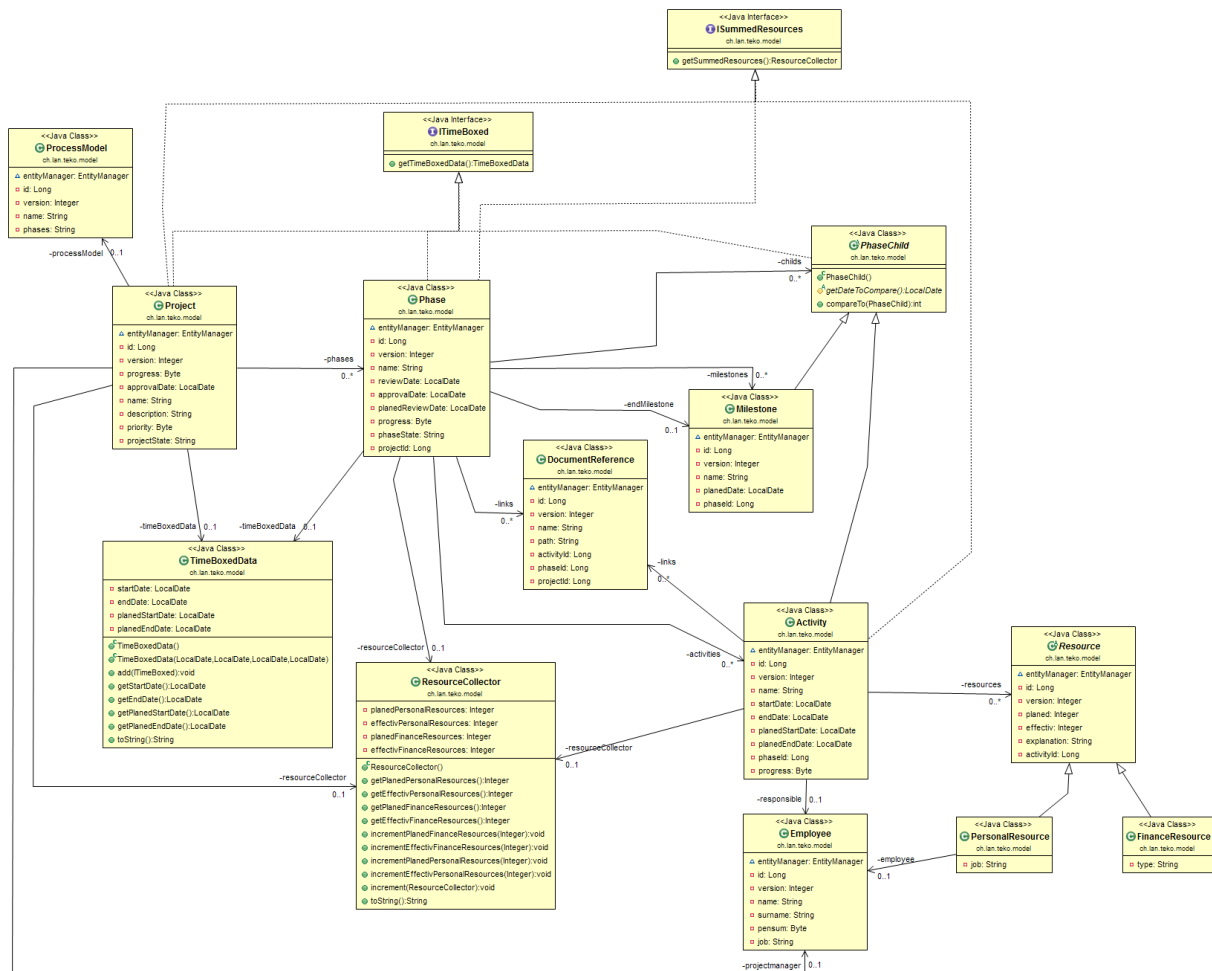
Die Fachklassen machen den deutlich grösseren Anteil aus. Sie haben alle gemeinsam, dass sie direkt in der Datenbank abgespeichert werden. Deshalb besitzen alle einen EntityManager welcher die Verbindung zur Datenbank darstellt. Er befindet sich in „javax.persistence.EntityManager“ und kommt aus dem Plugin „hibernate-jpa-2.1-api“. Eine detailliert Beschreibung findet man unter: <http://docs.oracle.com/javaee/6/api/javax/persistence/EntityManager.html>. Um die entsprechenden Fachklassen in der Datenbank zu speichern haben alle folgende drei Methoden:

- persist()
- merge()
- remove()

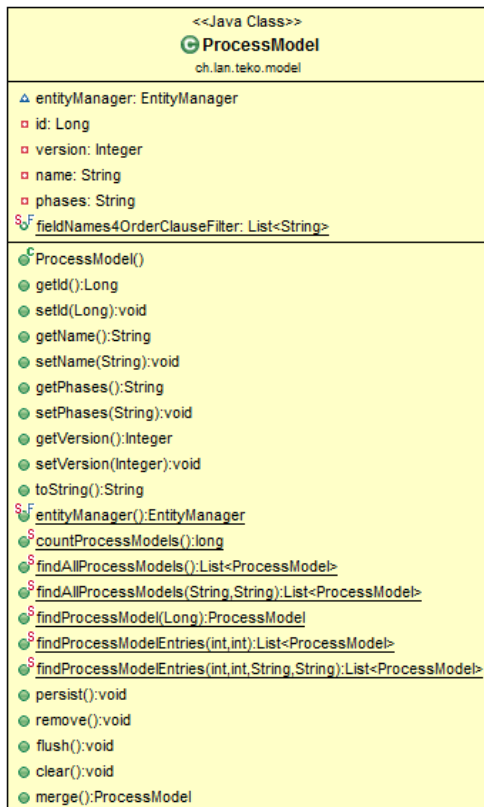
Diese speichern, aktualisieren oder löschen die Fachklassen aus der Datenbank. Anders als diese drei Methoden sind die Methoden um die Fachklassen aus der Datenbank zu laden alle statisch. Dies macht auch Sinn, weil man das entsprechende Objekt ja erst laden will. Wegen der Verbindung zur Datenbank haben alle Fachklassen eine Id, eine Version und eine statische Liste. Die Id repräsentiert der Primary-Key. Die Version wird vom EntityManager für ein Optimistic Locking verwendet (https://de.wikipedia.org/wiki/Optimistic_Concurrency). Die statische Liste wird für die Sortierung verwendet.

Wie der Name schon sagt, ist die Hauptaufgabe der Hilfsklassen die Fachklassen bei der Arbeit zu unterstützen.

Folgendes Klassendiagramm zeigt alle Klassen. Für die Übersichtlichkeit wurden bei den Fachklassen die Methoden weggelassen. Diese sind im entsprechenden Kapiteln der Klassen einzeln ersichtlich.

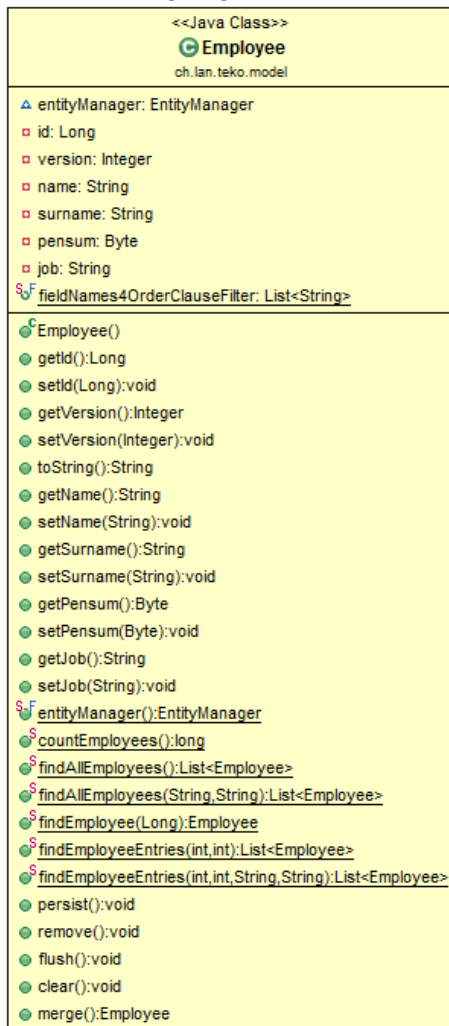


8.2.1 ProcessModel



Hiermit wird ein Vorgehensmodell abgebildet. Dadurch hat sie einen Namen und Phasen. Die Phasen werden, zu diesen Zeitpunkt im Projekt, in einem String gespeichert, getrennt durch ein Semicolon.

8.2.2 Employee



Hiermit wird ein Mitarbeiter abgebildet. Namen und Pensum dürften klar sein. Mit Job ist die Funktion gemeint, function kann man nicht mit allen Datenbanken verwenden. Deshalb wurde Job als Bezeichnung gewählt.

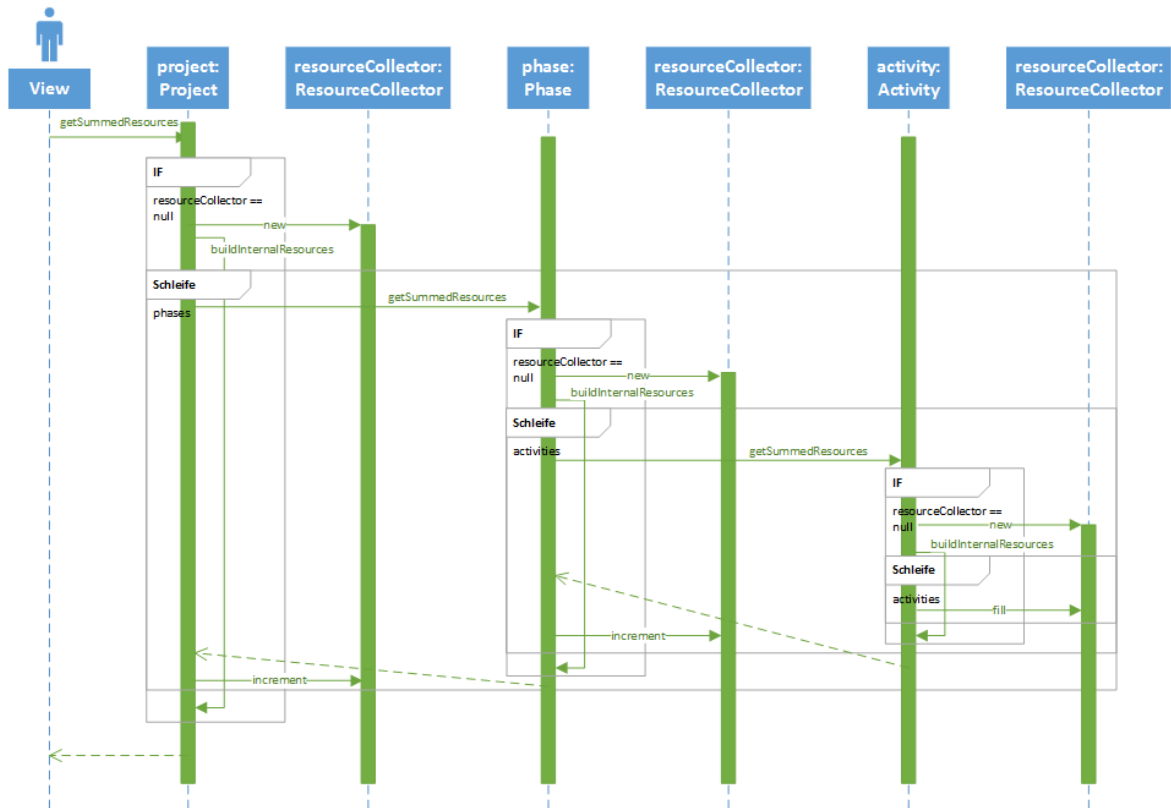
8.2.3 Project

<<Java Class>>  Project ch.lan.teko.model	
▲ entityManager: EntityManager S,F fieldNames4OrderClauseFilter: List<String> id: Long version: Integer progress: Byte approvalDate: LocalDate name: String description: String priority: Byte projectState: String projectmanager: Employee processModel: ProcessModel phases: List<Phase> resourceCollector: ResourceCollector timeBoxedData: TimeBoxedData	
Project() getSummedResources():ResourceCollector buildInternalResources():void toString():String getTimeBoxedData():TimeBoxedData buildTimeBoxedData():void addDocumentReference(Long,DocumentReference):void S,F entityManager():EntityManager S findProjectByPhaseId(Long):Project S findProjectByActivityId(Long):Project getProgress():Byte setProgress(Byte):void getApprovalDate():LocalDate setApprovalDate(LocalDate):void getName():String setName(String):void getDescription():String setDescription(String):void getPriority():Byte setPriority(Byte):void getProjectState():String setProjectState(String):void getProjectmanager():Employee setProjectmanager(Employee):void getProcessModel():ProcessModel setProcessModel(ProcessModel):void getPhases():List<Phase> setPhases(List<Phase>):void gettid():Long settid(Long):void getVersion():Integer setVersion(Integer):void S countProjects():long S findAllProjects():List<Project> S findAllProjects(String,String):List<Project> S findProject(Long):Project S findProjectEntries(int,int):List<Project> S findProjectEntries(int,int,String,String):List<Project> persist():void remove():void flush():void clear():void merge():Project	

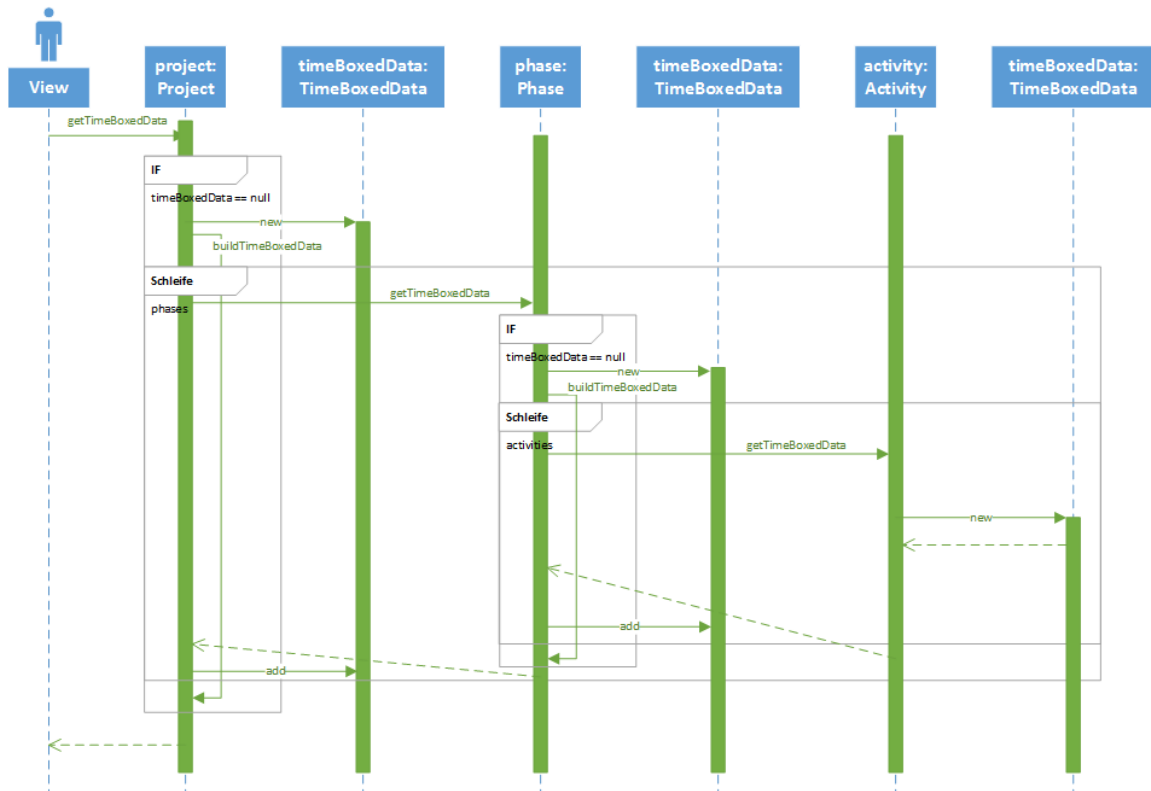
Hiermit wird ein Projekt abgebildet. Die meisten Attribute dürften klar sein, denn sie wurden aus den Anforderungen abgeleitet. Richtig interessant sind resourceCollector und timeBoxedData. Beide werden nicht in die Datenbank gespeichert sondern zur Laufzeit berechnet. In resourceCollector werden die Aufwände (PersonalResource & FinanceResource) für das gesamte Projekt zusammen

gezählt. In timeBoxedData wird die Dauer des Projekts berechnet. Wie das gemacht wird zeigen folgende Sequenzdiagramme.

Aufwände sammeln



Dauer berechnen

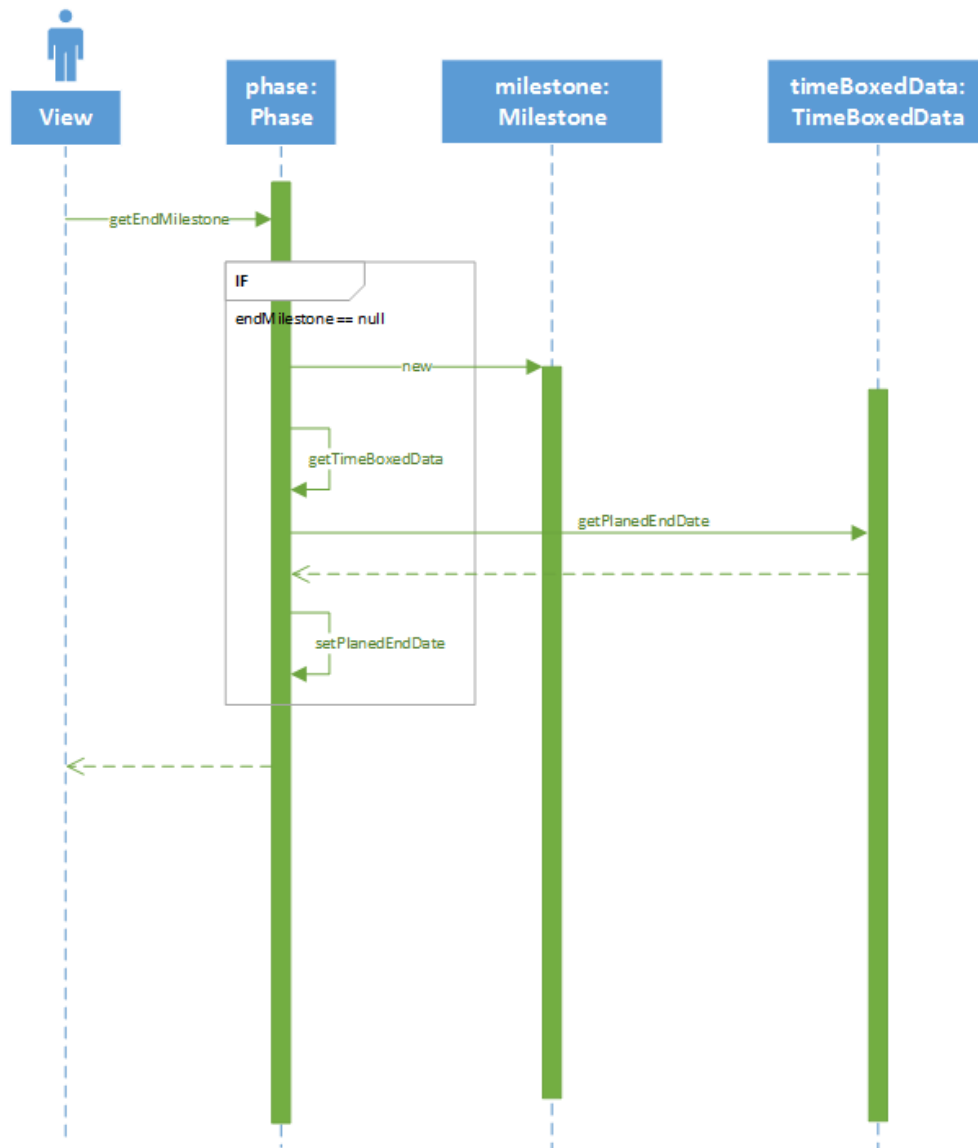


8.2.4 Phase

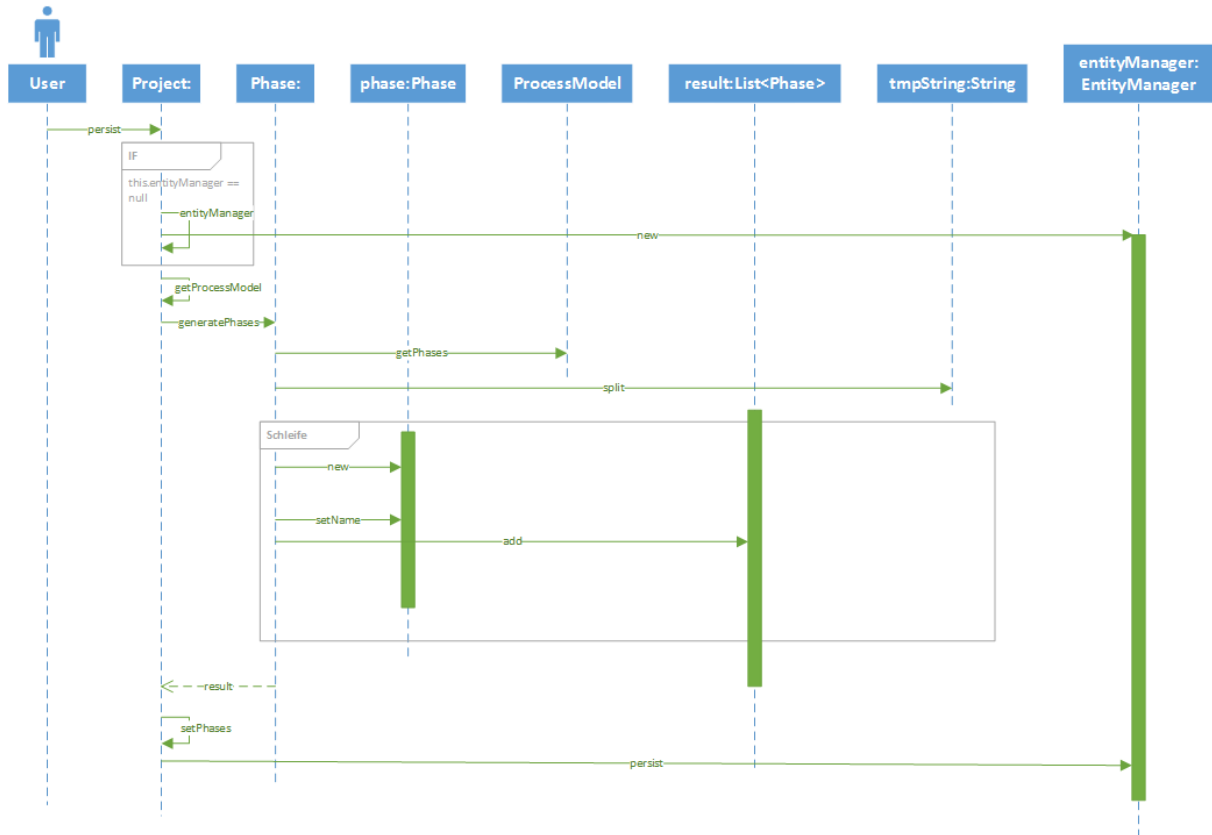
<<Java Class>> Phase ch.lan.teko.model	
▲ entityManager: EntityManager S,F fieldNames4OrderClauseFilter: List<String> □ id: Long □ version: Integer □ name: String □ links: Set<DocumentReference> □ reviewDate: LocalDate □ approvaDate: LocalDate □ planedReviewDate: LocalDate □ progress: Byte □ phaseState: String □ activities: Set<Activity> □ milestones: Set<Milestone> □ projectId: Long □ childs: SortedSet<PhaseChild> □ resourceCollector: ResourceCollector □ timeBoxedData: TimeBoxedData □ endMilestone: Milestone	
● Phase() ● getActivities():Set<Activity> ● addActivity(Activity):void ● removeActivity(Activity):void ● setActivities(Set<Activity>):void ● getMilestones():Set<Milestone> ● addMilestone(Milestone):void ● removeMilestone(Milestone):void ● setMilestones(Set<Milestone>):void ● getProjectId():Long ● setProjectId(Long):void ● getEndMilestone():Milestone ● toString():String ● getChilds():SortedSet<PhaseChild> ■ buildChilds():void ● getSummedResources():ResourceCollector ■ buildInternalResources():void ● getTimeBoxedData():TimeBoxedData ■ buildTimeBoxedData():void S generatePhases(ProcessModel):List<Phase> S addDocumentReference(Long,DocumentReference):void ● getId():Long ● setId(Long):void ● getVersion():Integer ● setVersion(Integer):void ● getLinks():Set<DocumentReference> ● setLinks(Set<DocumentReference>):void ● getReviewDate():LocalDate ● setReviewDate(LocalDate):void ● getApprovaDate():LocalDate ● setApprovaDate(LocalDate):void ● getPlanedReviewDate():LocalDate ● setPlanedReviewDate(LocalDate):void ● getProgress():Byte ● setProgress(Byte):void ● getPhaseState():String ● setPhaseState(String):void ● getName():String ● setName(String):void S,F entityManager():EntityManager S countPhases():long S findAllPhases():List<Phase> S findAllPhases(String,String):List<Phase> S findPhase(Long):Phase S findPhaseEntries(int,int):List<Phase> S findPhaseEntries(int,int,String,String):List<Phase> ● persist():void ● remove():void ● flush():void ● clear():void ● merge():Phase	

Hiermit wird eine Projektphase abgeleitet. Die meisten Attribute sind auch hier aus den Anforderungen abgeleitet und werden hier nicht näher beschrieben. Da auch die Phase eine Sammlung an Aufwänden und eine Dauer hat, findet man hier ebenfalls resourceCollector und timeBoxedData. Ihr Zweck und Verhalten ist gleich wie beim Projekt, nur halt auf Ebene der Phase. Erwähnenswert ist der endMilestone. Er wird zur Laufzeit berechnet und nicht in der Datenbank

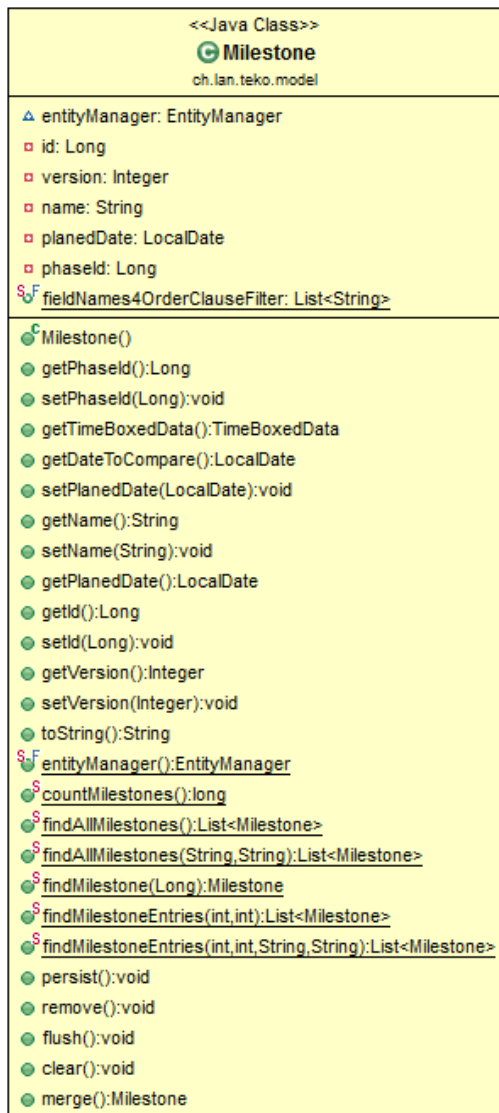
gespeichert. Sein Zweck ist es die Anforderung zu erfüllen, dass jede Phase mit einem Meilenstein endet. Sobald genügend Daten vorhanden sind um eine Dauer zu berechnen wird ein solcher Meilenstein erstellt. Hierzu folgendes Sequenzdiagramm.



Weiter gibt es eine statische Methode `generatePhases()` welche ein `ProcessModel` erwartet. Diese wird verwendet um, aufgrund eines Vorgehensmodells, Phasen zu erstellen. Dies geschieht immer beim Anlegen eines Projekts wie folgendes Sequenzdiagramm zeigt.



8.2.5 Milestone



Hiermit wird ein Meilenstein abgebildet. Diese Klasse ist genauso wie Activity von PhaseChild abgeleitet. Deshalb muss `getDateToCompare()` implementiert werden. Wozu die Klasse PhaseChild ist, steht im entsprechenden Kapitel. Hier wird `getDateToCompare()` so implementiert, dass `planedDate` zurückgegeben wird.

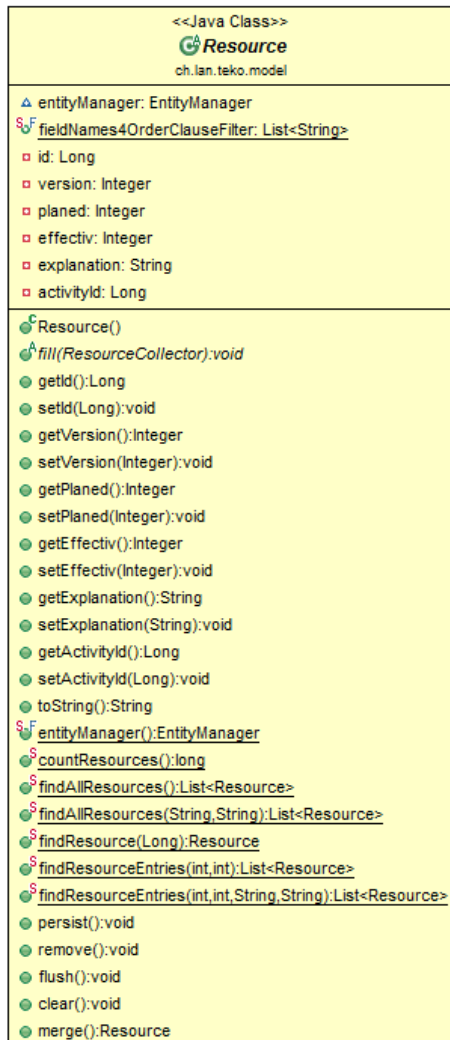
8.2.6 Activity

<<Java Class>> Activity ch.lan.teko.model
▲ entityManager: EntityManager S,F fieldNames4OrderClauseFilter: List<String> id: Long version: Integer name: String links: List<DocumentReference> startDate: LocalDate endDate: LocalDate planedStartDate: LocalDate planedEndDate: LocalDate resources: Set<Resource> responsible: Employee progress: Byte phaseld: Long resourceCollector: ResourceCollector personalResources: Set<PersonalResource> financeResources: Set<FinanceResource>
Activity() getDateToCompare():LocalDate compareTo(PhaseChild):int getSummedResources():ResourceCollector buildInternalResources():void getTimeBoxedData():TimeBoxedData S addDocumentReference(Long,DocumentReference):void getLinks():List<DocumentReference> setLinks(List<DocumentReference>):void getStartDate():LocalDate setStartDate(LocalDate):void getEndDate():LocalDate setEndDate(LocalDate):void getPlanedStartDate():LocalDate setPlanedStartDate(LocalDate):void getPlanedEndDate():LocalDate setPlanedEndDate(LocalDate):void getPhaseld():Long setPhaseld(Long):void getResources():Set<Resource> addResource(Resource):void removeResource(Resource):void setResources(Set<Resource>):void getId():Long setId(Long):void getVersion():Integer setVersion(Integer):void getResponsible():Employee setResponsible(Employee):void getProgress():Byte setProgress(Byte):void getName():String setName(String):void getPersonalResources():Set<PersonalResource> getFinanceResources():Set<FinanceResource> toString():String S,F entityManager():EntityManager S countActivities():long S findAllActivities():List<Activity> S findAllActivities(String,String):List<Activity> S findActivity(Long):Activity S findActivityEntries(int,int):List<Activity> S findActivityEntries(int,int,String,String):List<Activity> persist():void remove():void flush():void clear():void merge():Activity

Hiermit wird eine Aktivität abgebildet. Die zwei Arten von Ressourcen, PersonalResource und FinanceResource, werden über die gemeinsame Super-Klasse in einer HashSet gespeichert. Das sah beim Design gut aus, bereitete in der UI aber Probleme. Deshalb gibt es zwei zusätzliche Sets (personalResources und financeResource) beide werden erst zur Laufzeit gefüllt und nicht

abgespeichert. Würde das Projekt länger gehen, wäre dies eine der ersten Stellen die umgebaut würden. Die getDateToCompare() Methode wird hier so implementiert, dass planedStartDate zurückgegeben wird. Mehr dazu unter PhaseChild.

8.2.7 Resource



Hiermit werden die Gemeinsamkeiten von **PersonalResource** und **FinanceResource** abgebildet. Auffällig ist die abstrakte Methode **fill()** welche einen **ResourceCollector** erwartet. Damit wird sichergestellt, dass Aufwände von Ableitungen immer über einen **ResourceCollector** gesammelt werden können.

8.2.8 PersonalResource

<<Java Class>>	
PersonalResource	
ch.lan.teko.model	
<ul style="list-style-type: none"> job: String employee: Employee 	
<u>fieldNames4OrderClauseFilter: List<String></u>	
<ul style="list-style-type: none"> PersonalResource() fill(ResourceCollector):void toString():String countPersonalResources():long findAllPersonalResources():List<PersonalResource> findAllPersonalResources(String,String):List<PersonalResource> findPersonalResource(Long):PersonalResource findPersonalResourceEntries(int,int):List<PersonalResource> findPersonalResourceEntries(int,int,String,String):List<PersonalResource> merge():PersonalResource getJob():String setJob(String):void getEmployee():Employee setEmployee(Employee):void 	

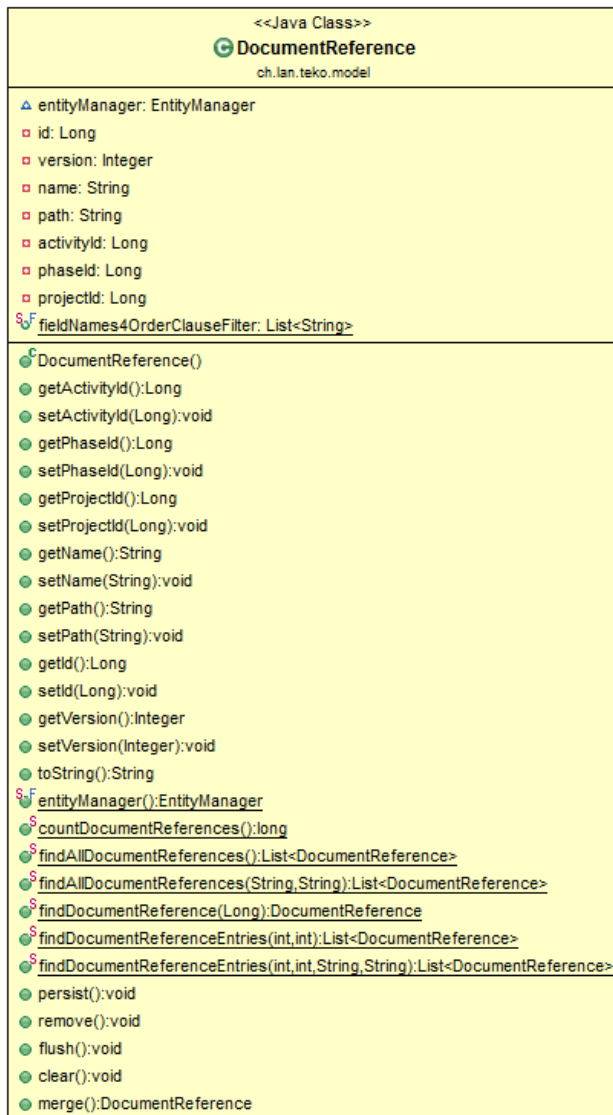
Hiermit werden Personenaufwände abgebildet. Die fill() Methode ist so implementiert, dass die Personenaufwände spezifisches Daten im ResourceCollector gefüllt werden.

8.2.9 FinanceResource

<<Java Class>>	
FinanceResource	
ch.lan.teko.model	
<ul style="list-style-type: none"> type: String 	
<u>fieldNames4OrderClauseFilter: List<String></u>	
<ul style="list-style-type: none"> FinanceResource() fill(ResourceCollector):void getType():String setType(String):void countFinanceResources():long findAllFinanceResources():List<FinanceResource> findAllFinanceResources(String,String):List<FinanceResource> findFinanceResource(Long):FinanceResource findFinanceResourceEntries(int,int):List<FinanceResource> findFinanceResourceEntries(int,int,String,String):List<FinanceResource> merge():FinanceResource toString():String 	

Hiermit werden Externe Kosten abgebildet. Die fill() Methode ist so implementiert, dass die „Externen Kosten Aufwände“ spezifischen Daten im ResourceCollector gefüllt werden.

8.2.10 DocumentReference



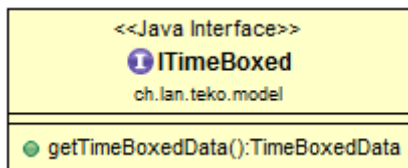
Hiermit werden die Dokumentreferenzen abgebildet. Diese Klasse hat nichts außergewöhnliches, was Gelegenheit bietet die Methoden `flush()` und `clear()` zu erläutern. Man findet diese auch in den Fachklassen. Bei `flush()` handelt es sich um das Pendant zu einem Commit in einer Datenbank. `clear()` hingegen löscht alle Änderungen welche nicht mit einem `flush()` zur Datenbank gesendet worden sind. Ausserdem gibt es drei Attribute `activityId`, `phaseId` und `projectId` auch diese findet man in anderen Fachklassen. Wenn man genauer hinsieht verbirgt sich dahinter immer die entgegengesetzte Richtung einer `OneToMany` Beziehung. Da die `DocumentReference` in `Project`, `Phase` und `Activity` eine `ManyToOne` Beziehung hat, sind hier alle drei vertreten.

8.2.11 ISummedResources



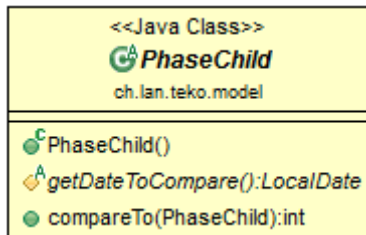
Die erste Hilfsklasse in der Aufzählung, wobei eigentlich ist es nur ein Interface. Alle Ableitungen sollen einen `ResourceCollector` zur Verfügung stellen. Dieser kann bereits gefüllt sein.

8.2.12 ITimeBoxed



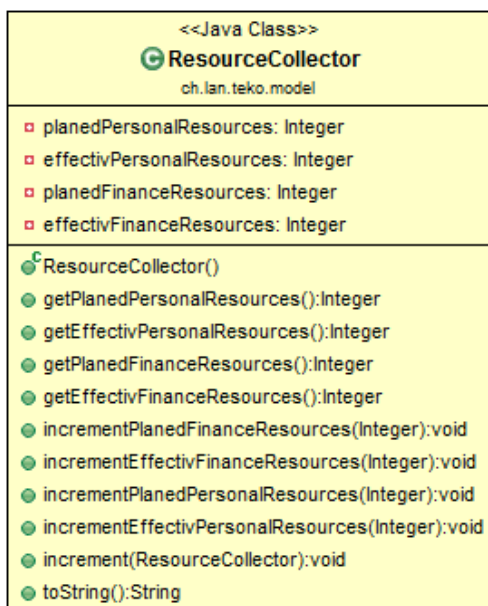
Zweites „Hilfs-Interface“. Alle Ableitungen sollen ein `TimeBoxedData` zur Verfügung stellen. Dieser kann bereits gefüllt sein.

8.2.13 PhaseChild



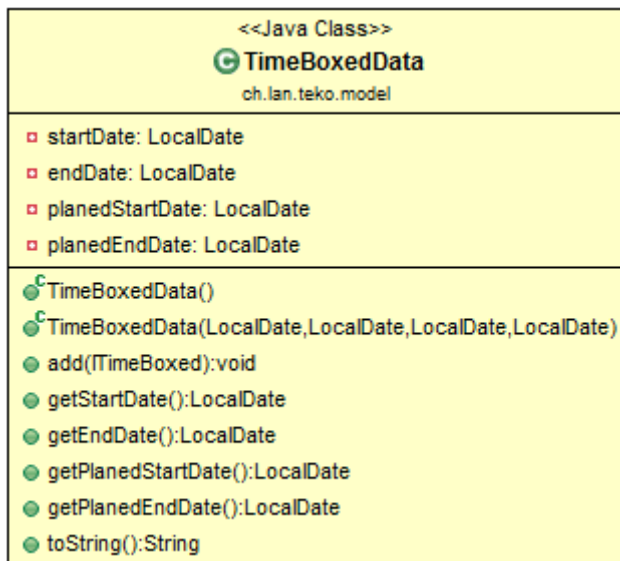
Hilfsklasse für die Sortierung von „Kinder“ einer Phase (bisher Milestone und Activity). Der Inhalt einer Phase soll im UI nach einem Datum sortiert angezeigt werden. Dazu implementiert diese Klasse das `Comparable` Interface und verlangt von seinen Ableitungen über `getDateToCompare()` ein Datum zum Sortieren.

8.2.14 ResourceCollector



Hilfsklasse um die Aufwände zusammen zu zählen. Die Attribute entsprechen den möglichen Aufwänden einmal als Geplant und einmal als Effektiv. Die Aufwände können entweder separat über die `increment...()` Methoden hochgezählt werden oder es werden gleich die Aufwände von einem anderen `ResourceCollector` hinzugefügt. So ist es einfach möglich die Aufwände für ein gesamtes Projekt zu sammeln. Siehe das „Aufwände sammeln“ Sequenzdiagramm unter Project.

8.2.15 TimeBoxedData

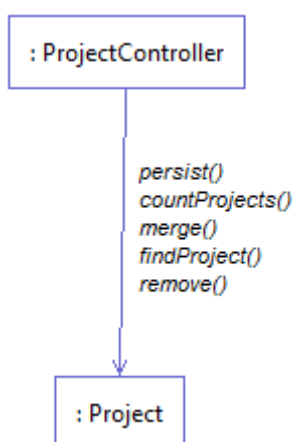


Hilfsklasse um die Dauer zu berechnen. Die Attribute zeigen eine Geplante und eine Effektive Dauer. Die Attribute können entweder mit Hilfe des Konstruktors gesetzt werden oder über die `add()` Methode. Die `add()` Methode nimmt jeweils das früheste Start und späteste End Datum und merkt sich diese. Damit ist es ähnlich wie beim `ResourceCollector` einfach die gesamte Dauer eines Projekts zu berechnen. Siehe das „Dauer berechnen“ Sequenzdiagramm unter Project.

8.3 Controller

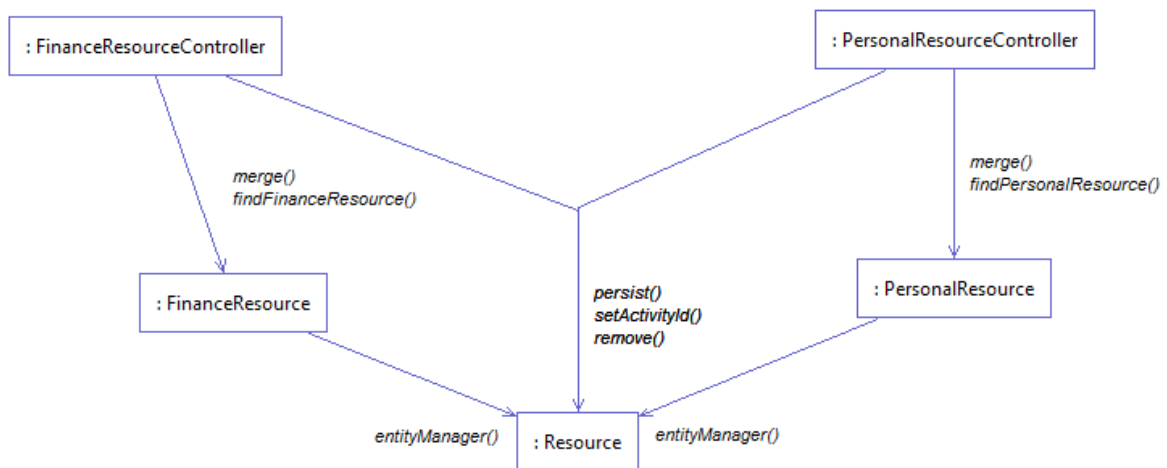
Die Verantwortung der Controller besteht darin eine Benutzer Anfrage entgegen zu nehmen, im Model die entsprechende Logik auf zu rufen und dann auf eine View zu verweisen.

Grundsätzlich gibt es für jede Fachklasse einen Controller welcher die Create, Read, Update und Delete (CRUD) Funktionen bereitstellt. Hier stellvertretend mit dem Project als Kommunikationsdiagramm dargestellt.



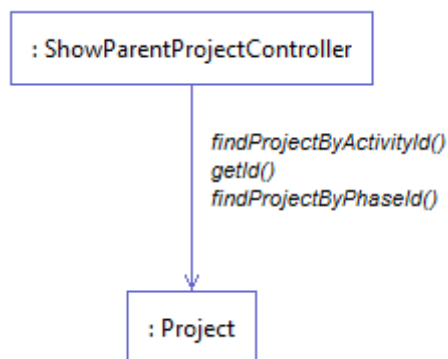
So oder so ähnlich sehen alle Diagramme zwischen Controller und Fachklasse aus. Es gibt jedoch zwei besondere Fälle, die hier weiter beschrieben werden.

Für `PersonalResource` und `FinanceResource` wird zum Anlegen und Löschen nicht die entsprechende Fachklasse aufgerufen sondern nur `Resource`. Folgendes Diagramm zeigt dies.



Man sieht auch gleich, dass sich beide Fachklassen den EntityManager teilen und somit auch die Controller gleich auf Resource zugreifen können.

Der ShowParentProjectController wird verwendet um in der UI einen Link auf das übergeordnete Projekt bereit zu stellen. Deshalb sucht er sich je nach Aufrufer das Projekt und macht eine Weiterleitung an den ProjectController. Folgendes Diagramm zeigt die möglichen Such-Aufrufe auf das Projekt.



8.4 View

Die Verantwortung der View besteht darin dem Benutzer die Daten sowie eine Oberfläche um Daten zu ändern anzuzeigen. Da es sich hierbei um eine Webapplikation handelt welche das UI im Browser anzeigt wurde das Design in CSS Datei ausgelagert. Hier wird auf eine Vorlage von Bootstrap gesetzt. Um z.B. Validierungen direkt im Client zu machen werden Javascript Frameworks von Bootstrap und JQuery eingesetzt.


Die eigentlichen Ansichten auf bestimmte Daten sind im jspx Dateien beschrieben. Darin ist eine Mischung aus HTML und JSP enthalten. Dadurch ist es möglich Java-Code und spezielle JSP-Aktionen in HTML-Seiten einzubetten.

Der Aufbau jeder Seite ist gleich gehalten. Zu Oberst befindet sich eine Navigation mit der man zu den Projekten, Mitarbeitern und Vorgehensmodellen gelangt. Hat man einmal ein Projekt, eine Phase oder eine Aktivität ausgewählt erscheinen links Informationen dazu. In der Mitte wird eine Maske entweder zum Anlegen, Bearbeiten oder Anzeigen eines Objekts dargestellt.



Im Folgenden sind die wichtigsten Seiten kurz beschrieben.

8.4.1 Projektübersicht


[projects](#)
[employees](#)
[processmodels](#)

Projectmanagement

Project

Name:	LernToRead
Progress:	1
Approval Date:	01.01.2016
Description:	
Priority:	10
Project State:	Start
Projectmanager:	Hans Peterson PM
Process Model:	ABC Model
Geplanter Start:	2016-02-02
Geplantes Ende:	2016-04-10
Geplante Externe-Kosten:	10
Effektive Externe-Kosten:	50
Geplanter Personenaufwand:	100
Effektiver Personenaufwand:	8

[Edit](#)

Phase A

Geplant: 2016-02-02 bis 2016-02-10

Actions

FindSomeoneWhoCanRead

Geplant: 2016-02-02 bis 2016-02-10

Actions

Abschluss von Phase A

Geplant: 2016-02-10

Phase B

Geplant: 2016-03-02 bis 2016-03-10

Actions

ReadingBooks

Geplant: 2016-03-02 bis 2016-03-10

Actions

Abschluss von Phase B

Geplant: 2016-03-10

Phase C

Geplant: 2016-04-02 bis 2016-04-10

Actions

Reading heavy stuff

Geplant: 2016-04-02 bis 2016-04-10

Actions

Personal effort

Funktion: Reader, Geplant: 90std, Effektiv: 4std, Mitarbeiter: Chef Wiggum

Funktion: PM, Geplant: 10std, Effektiv: 4std, Mitarbeiter: Hans Peterson

External costs

Kostenstelle: Büro, Geplant: 10std, Effektiv: 50std


Abschluss von Phase C

Geplant: 2016-04-10

Sprache:  

Hier sieht man die Übersicht zu einem Ausgewählten Projekt. Links werden die Projekte spezifischen Daten angezeigt. In der Mitte sieht man die Phase mit ihren Aktivitäten und Meilensteinen. Innerhalb einer Aktivität sieht man die Aufwände in zwei Kategorien unterteilt.

8.4.2 Phase bearbeiten


[projects](#)
[employees](#)
[processmodels](#)

Projectmanagement

Project

Name:	LernToRead
Progress:	1
Approval Date:	01.01.2016
Description:	
Priority:	10
Project State:	Start
Projectmanager:	Hans Peterson PM
Process Model:	ABC Model
Geplanter Start:	2016-02-02
Geplantes Ende:	2016-04-10
Geplante	190
Externe-Kosten:	
Effektive	52
Externe-Kosten:	
Geplanter	100
Personenaufwand:	
Effektiver	8
Personenaufwand:	

Phase aktualisieren

Name:	<input type="text" value="Phase A"/>
Review Date:	<input type="text"/>
Approval Date:	<input type="text"/>
Planned Review Date:	<input type="text"/>
Progress:	<input type="text"/>
Phase State:	<input type="text"/>

Speichern

Liste alle Activitys

Name	Start Date	End Date	Planned Start Date	Planned End Date			
FindSomeoneWhoCanRead			02.02.2016	10.02.2016			

Activity erstellen

Liste alle Milestones

Name	Planned Date			
Erste Hürde	13.03.2016			

Milestone erstellen

Liste alle Document References


Name	Path			
TestFile	C:\Semesterarbeit\Projektverwaltung\TestFile.pdf			

Document Reference erstellen

Sprache:

Diese Ansicht zeigt die Maske um eine Phase zu bearbeiten. Im oberen Teil sieht man die für die Phase relevanten Eingabefelder. Im unteren Teil sieht man die Aktivitäten, Meilensteine und Dokumentreferenzen. Links werden die Daten des übergeordneten Projekts angezeigt. Klickt man auf Project gelangt man wieder zur Projektübersicht.

8.4.3 Aktivität bearbeiten


[projects](#)
[employees](#)
[processmodels](#)

Projectmanagement

Phase







Name: Phase C
Review Date:
Approval Date:
Planned Review Date:
Progress:
Phase State:
Geplanter Start: 2016-04-02
Geplantes Ende: 2016-04-10
Geplante 100
Externe-Kosten:
Effektive 50
Externe-Kosten:
Geplanter 100
Personenaufwand:
Effektiver 8
Personenaufwand:

Activity aktualisieren

Name: Reading heavy stuff
Start Date:
End Date:
Planned Start Date: 02.04.2016
Planned End Date: 10.04.2016
Responsible: Chef Wiggum Tester
Progress: 0







Speichern

Liste alle Personal Resources

Job	Employee	Planned	Effectiv	Explanation			
PM	Hans Peterson PM	10	4				
Reader	Chef Wiggum Tester	90	4				

Personal Resource erstellen

Liste alle Finance Resources



Type	Planned	Effectiv	Explanation			
Büro	10	50	ups			
Büro	90	0	abe gheit			

Finance Resource erstellen

Liste alle Document References

Document Reference konnte nicht gefunden werden.

Document Reference erstellen

Sprache:  

Hier sieht man die Ansicht zum Bearbeiten einer Aktivität. Wieder erst die Daten für die Aktivität selbst. Darunter drei Listen, die erste mit den Personenaufwänden, die zweite mit den Externen-

Kosten und die dritte mit den Dokumentenreferenzen. Links sieht man nun die Daten der übergeordneten Phase. Mit einem Klick auf Phase gelangt man direkt zur Projektübersicht.

8.4.4 Personenaufwand bearbeiten

Activity

Name:	Reading heavy stuff
Start Date:	
End Date:	
Planned Start Date:	02.04.2016
Planned End Date:	10.04.2016
Progress:	0
Responsible:	Chef Wiggum Tester
Geplanter Start:	2016-04-02
Geplantes Ende:	2016-04-10
Geplante	100
Externe-Kosten:	
Effektive	50
Externe-Kosten:	
Geplanter	100
Personenaufwand:	
Effektiver	8
Personenaufwand:	

Personal Resource aktualisieren

Planned:

Effectiv:

Explanation:

Job:

Employee:

Speichern

Sprache:

Diese Ansicht zeigt das Bearbeiten eines Personenaufwandes. Die Ansicht zum Bearbeiten Externer-Kosten sieht fast identisch aus. Der einzige Unterschied besteht in den unterschiedlichen Eingabefeldern. Links sieht man die übergeordnete Aktivität. Mit einem Klick auf Activity gelangt man auf die Projektübersicht.

8.5 Datenbank

Für den Prototypen wird auf eine In-Memory Datenbank gesetzt. Das Mapping zwischen Java und der Datenbank wird mit Hibernate und JPA realisiert. Hibernate und JPA übernehmen weiter auch den Aufbau des Datenbankschemas, deshalb wird es hier nicht näher beschrieben. Bei einer Änderung des Datenmodells wird die Datenbank automatisch angepasst. Allfällige Datenmigrationen werden nicht unterstützt.

8.6 Code

8.6.1 Java

Die Java-Klassen sind je nach ihrer Funktion in einem dieser packages zu finden.

```

src/main/java
├── ch.lan.teko.controller
├── ch.lan.teko.model
└── ch.lan.teko.util
  
```

Die Unit-Tests befinden sich im gleichen package aber unter src/test/java.

8.6.1.1 Kommentar

Kommentar wird sparsam eingesetzt, da guter Code sich selbst kommentiert und mit Kommentar nur unübersichtlich wird. Es soll jedoch jede Klasse einen Klassenkommentar haben, welche die Klasse kurz vorstellt. Weiter sollen alle „nicht private“ Methoden und Attribute ebenfalls mit Javadoc kommentiert werden. Dies deshalb weil man eh nur „nicht private“ Member von Aussen verwenden darf und man sich damit die Klasse nicht anschauen muss.

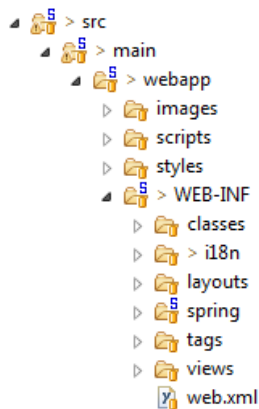
8.6.2 Konfiguration

Die Konfiguration befindet sich unter `src/main/resources`. Informationen zum Spring-Context stehen in `applicationContext.xml`. Informationen zur Datenbank in den `database.properties`. Die Log-Konfiguration steht in den `log4j.properties`.

Für Tests gibt es bisher keine extra Konfiguration. Sobald eine benötigt wird, findet man sie unter `src/test/resources`.

8.6.3 Layout

Der Code rund ums UI hat folgenden Aufbau:



Images, scripts und styles dürfte klar sein. Interessant sind die Verzeichnisse innerhalb von WEB-INF. Classes und Layouts wird gebraucht und verschiedene Layouts anzuzeigen, dies wird mit der jetzigen Version jedoch nicht unterstützt. I18n enthält die Übersetzungen. In spring steht die Konfiguration von Spring-MVC. In tags sind hilfreiche Anzeige-Funktionen wie z.B. eine Tabelle mit Werten anzeigen. Unter views verbergen sich alle Anzeige-Seiten der Applikation.

9 Einführung

Damit die Applikation betrieben werden kann sind folgende Kriterien zu erfüllen.

9.1 Server

	Beschreibung
Hardware	Mindestens zwei Kerne mit je 2.4 GHz, 8GB Arbeitsspeicher und 10 GB Festplattenkapazität
Betriebssystem	Mindestens Debian 6.0 oder Windows 7
Java-Version	Mindestens Java 8
Web-Server	Mindestens Apache Tomcat 7.0.67

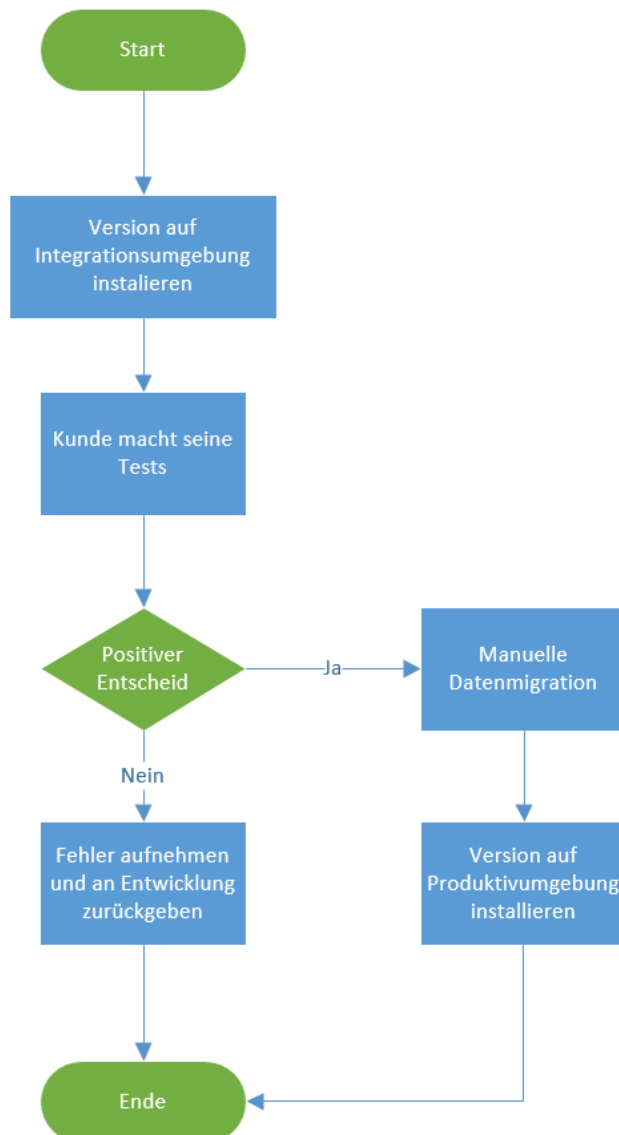
9.2 Client

Die Clients müssen über das Netzwerk Zugriff auf den Server haben.

Beschreibung	
Hardware	Mindestens zwei Kerne mit je 2.4 GHz, 4GB Arbeitsspeicher
Betriebssystem	Mindestens Ubuntu 14.04 oder Windows 7
Unterstützte Browser	Firefox ab Version 44

9.3 Einführungsprozess

Jede Version die eingeführt wird durchläuft folgenden Prozess.



10 Qualitätsmanagement

10.1 Testkonzept

10.1.1 Ziel

Ziel eines Tests ist es z.B. einen Anwendungsfall so zu prüfen, dass man davon ausgehen kann, dass er unter allen Voraussetzungen einwandfrei funktioniert.

10.1.2 Testfall-Analyse

Folgende Systemanwendungsfälle stellen durch ihre hohe Priorität ein Risiko dar und müssen zwingend durch Testfälle abgedeckt werden.

- Neues Projekt anlegen
- Mitarbeiter auswählen
- Vorgehenmodell auswählen
- Phasen generieren
- Neues Vorgehenmodell anlegen
- Neue Mitarbeiter anlegen
- Aktivitäten verwalten
- Meilensteine verwalten
- Ressourcen verwalten

Daraus ergeben sich folgende Testfälle:

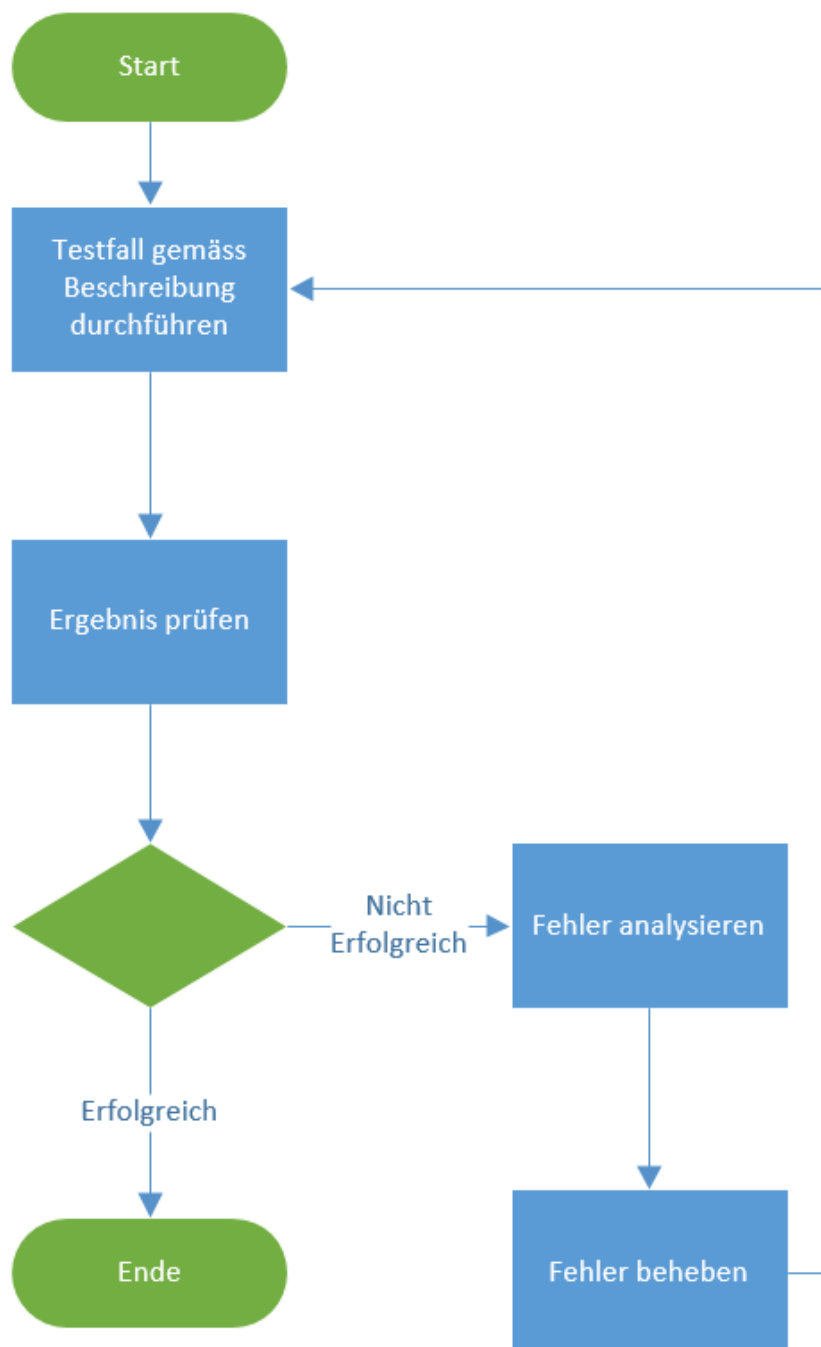
Testfall	Systemanwendungsfälle
Mitarbeiter verwalten	Mitarbeiter anlegen <i>*Mitarbeiter bearbeiten</i>
Vorgehensmodell verwalten	Neues Vorgehenmodell anlegen <i>*Bestehendes Vorgehensmodell bearbeiten</i>
Projekt verwalten	Neues Projekt anlegen Mitarbeiter auswählen Vorgehenmodell auswählen Phasen generieren <i>*Projektmanagement spezifische Informationen verwalten</i>
Aktivität verwalten	Aktivität verwalten
Personenaufwand verwalten	Ressourcen verwalten
Externe-Kosten verwalten	Ressourcen verwalten

**Diese Systemanwendungsfälle haben zwar eine geringere Priorität wurden aber aus praktischen Gründen in den Testfall integriert.*

Die Systemanwendungsfälle „Dokumentreferenzen verwalten“ & „Meilensteine verwalten“ haben eine geringere Priorität, für eine vollfunktionsfähige Applikation sind sie dennoch wichtig. Deshalb wurden daraus folgende Testfälle erstellt.

Testfall	Systemanwendungsfälle
Meilensteine verwalten	Meilensteine verwalten
Dokumentreferenzen verwalten	Dokumentreferenzen verwalten

10.1.3 Testprozess



10.1.4 Testarten

Nr.	Testart	Beschreibung
1	Unittests	Für jede Klasse wird ein Unittest erstellt. Ausnahmen bilden nur Klassen die keine Logik beinhalten z.B. einfach Container-Klassen. Im Unittest werden nur die Funktionen der zu testenden Klasse überprüft, andere benötigten Funktionen werden allenfalls gemockt.
2	Manuelle Tests nach Testplan	Die Testfälle werden von einem Tester ausgeführt.

10.1.5 Fehlerklassen

Nr.	Fehlerklasse	Beschreibung
1	Fehlerfrei	Es wurden keine Fehler entdeckt. Das getestete Modul bzw. System arbeitet 100% fehlerfrei.
2	Minor	Es wurden Fehler entdeckt, welche aber den Betrieb der Software nicht stören. Fehler können z. B. Schreibfehler sein.
3	Major	Es wurden Fehler entdeckt, welche noch nicht als kritisch angesehen werden müssen. Diese Fehler können aber nicht bestehen und müssen in absehbarer Zeit behoben werden.
4	Critical	Es wurden Fehler entdeckt, welchen den Betrieb der Software nicht mehr sicherstellen. Es sind Fehler die nicht heute auf morgen gelöst werden können und nochmals eine intensive Planung im fehlerhaften Modul herbeiführen.
5	Blocker	Es wurden Fehler entdeckt, welche den Betrieb der Software verunmöglichen. Solche Fehler müssen mit grösster Priorität angegangen werden. Sie können das unter Umständen das ganze Projekt in Gefahr bringen.

10.1.6 Testfälle

10.1.6.1 Mitarbeiter verwalten

Id / Titel	T-01 / Mitarbeiter verwalten
Priorität	1
Beschreibung	Ein Mitarbeiter mit folgenden Daten anlegen: Vorname: Hans Name: Peterson Pensum: 50 Job: Tester Den Namen in T01 ändern
Testvoraussetzung	Applikation ist gestartet
Testschritte	-Ansicht: neuer Mitarbeiter öffnen -Daten eingeben -Mitarbeiter speichern -Daten überprüfen -Namen ändern -Speichern -Namen überprüfen
Erwartetes Ergebnis	Mitarbeiter wird mit eingegebenen Daten angezeigt. Der geänderte Name wurde übernommen.

10.1.6.2 Vorgehensmodell verwalten

Id / Titel	T-02 / Vorgehensmodell verwalten
Priorität	1
Beschreibung	Ein Vorgehensmodell mit folgenden Daten anlegen: Titel: VM01 Phasen: P1, P2, P3 Den Titel in T02 ändern.
Testvoraussetzung	Applikation ist gestartet
Testschritte	-Ansicht: neues Vorgehensmodell öffnen -Daten eingeben

	-Vorgehensmodell speichern -Daten überprüfen -Titel ändern -Speichern -Titel überprüfen
Erwartetes Ergebnis	Vorgehensmodell wird mit eingegebenen Daten angezeigt. Der geänderte Titel wurde übernommen.

10.1.6.3 Projekt verwalten

Id / Titel	T-03 / Projekt verwalten
Priorität	1
Beschreibung	Ein Projekt mit folgenden Daten anlegen: Name: Projekt1 Progress: 0 Priority: 1 Project State: Start Mitarbeiter: Hans T01 Vorgehensmodell: T02 Den Namen in T03 ändern.
Testvoraussetzung	Applikation ist gestartet Der Mitarbeiter Hans T01 und das Vorgehensmodell T02 sind erfasst
Testschritte	-Ansicht: neues Projekt öffnen -Daten eingeben -Projekt speichern -Daten überprüfen -Namen ändern -Speichern -Namen überprüfen
Erwartetes Ergebnis	Projekt wird mit eingegebenen Daten angezeigt. Die Phasen aus dem Vorgehensmodell wurden generiert. Der geänderte Name wurde übernommen.

10.1.6.4 Aktivität verwalten

Id / Titel	T-04 / Aktivitäten verwalten
Priorität	1
Beschreibung	Eine Aktivität mit folgenden Daten anlegen: Den Titel in T04 ändern
Testvoraussetzung	Ein Projekt ist erfasst
Testschritte	-Ansicht: neue Aktivität anlegen -Daten eingeben -Aktivität speichern -Daten überprüfen -Titel ändern -Speichern -Titel überprüfen
Erwartetes Ergebnis	Aktivität wird mit eingegebenen Daten angezeigt. Der geänderte Titel wurde übernommen.

10.1.6.5 Dokumentreferenzen verwalten

Id / Titel	T-05 / Dokumentreferenzen verwalten
Priorität	2
Beschreibung	<p>Eine Dokumentreferenz mit folgenden Daten anlegen:</p> <p>Name: Ref01</p> <p>Pfad: C:/tmp/doc.txt</p> <p>Den Namen in T05 ändern.</p> <p>Die Dokumentenreferenz löschen.</p>
Testvoraussetzung	Ein Projekt ist erfasst
Testschritte	<ul style="list-style-type: none"> -Ansicht: neue Dokumentreferenz -Daten eingeben -Dokumentreferenz speichern -Daten überprüfen -Namen ändern -Speichern -Namen überprüfen -Löschen -Prüfen ob die Referenz nicht mehr vorhanden ist
Erwartetes Ergebnis	Dokumentenreferenz wird mit eingegebenen Daten angezeigt. Der geänderte Name wurde übernommen. Nach dem Löschen ist der Meilenstein nicht mehr vorhanden.

10.1.6.6 Meilensteine verwalten

Id / Titel	T-06 / Meilensteine verwalten
Priorität	2
Beschreibung	<p>Ein Meilensteine mit folgenden Daten anlegen:</p> <p>Name: M01</p> <p>Phase: P2</p> <p>Datum: ein Tag vor dem Ende von P2</p> <p>Den Namen in T06 ändern.</p> <p>Meilenstein löschen</p>
Testvoraussetzung	Eine Phase P2 ist erfasst. Sie dauert min 2 Tage.
Testschritte	<ul style="list-style-type: none"> -Ansicht: neuer Meilenstein anlegen -Daten eingeben -Meilenstein speichern -Daten überprüfen -Namen ändern -Speichern -Namen überprüfen -Löschen -Prüfen ob Meilenstein nicht mehr vorhanden ist
Erwartetes Ergebnis	Meilenstein wird mit eingegebenen Daten angezeigt. Der geänderte Name wurde übernommen. Nach dem Löschen ist der Meilenstein nicht mehr vorhanden.

10.1.6.7 Personenaufwand verwalten

Id / Titel	T-07 / Personenaufwand verwalten
Priorität	1
Beschreibung	Einen Personenaufwand mit folgenden Daten anlegen: Geplant: 100 Effektiv: 50 Job: Testen Mitarbeiter: T01 Den Job in T07 ändern. Personenaufwand löschen
Testvoraussetzung	Eine Aktivität ist erfasst, es gibt den Mitarbeiter T01.
Testschritte	-Ansicht: neuer Personenaufwand hinzufügen -Daten eingeben -Speichern -Daten überprüfen -Job ändern -Speichern -Job überprüfen -Löschen -Prüfen ob Aufwand nicht mehr vorhanden ist
Erwartetes Ergebnis	Personenaufwand wird mit eingegebenen Daten angezeigt. Der geänderte Job wurde übernommen. Nach dem Löschen ist der Aufwand nicht mehr vorhanden.

10.1.6.8 Externe-Kosten verwalten

Id / Titel	T-08 / Externe-Kosten verwalten
Priorität	1
Beschreibung	Einen Externen-Kostenpunkt mit folgenden Daten anlegen: Geplant: 100 Effektiv: 50 Typ: Testen Den Typ in T08 ändern. Kostenpunkt löschen
Testvoraussetzung	Eine Aktivität ist erfasst.
Testschritte	-Ansicht: neuer Externe-Kosten hinzufügen -Daten eingeben -Speichern -Daten überprüfen -Typ ändern -Speichern -Typ überprüfen -Löschen -Prüfen ob Kosten nicht mehr vorhanden ist
Erwartetes Ergebnis	Externen-Kostenpunkt wird mit eingegebenen Daten angezeigt. Der geänderte Job wurde übernommen. Nach dem Löschen ist sind Kosten nicht mehr vorhanden.

10.1.7 Testprotokolle

10.1.7.1 Testprotokoll vom 22.01.16

Testfall Id	T01
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T02
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T03
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

10.1.7.2 Testprotokoll vom 29.01.16

Testfall Id	T01
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T02
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T03
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T04
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T05
Resultat	Nicht OK
Bemerkungen	Das Löschen hat zu einem Fehler geführt. Problem wurde gefunden und behoben.
Fehlerklasse	3

Testfall Id	T06
Resultat	Nicht OK
Bemerkungen	Das Löschen hat zu einem Fehler geführt. Problem wurde gefunden und behoben.
Fehlerklasse	3

10.1.7.3 Testprotokoll vom 12.02.16

Testfall Id	T01
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T02
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T03
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T04
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T05
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T06
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T07
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T08
Resultat	OK

Bemerkungen	-
Fehlerklasse	1

10.1.7.4 Testprotokoll vom 25.02.16

Testfall Id	T01
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T02
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T03
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T04
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T05
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T06
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T07
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

Testfall Id	T08
Resultat	OK
Bemerkungen	-
Fehlerklasse	1

11 Konfigurationsmanagement

11.1 Hardware

Die Dokumentation und Applikation werden auf einem HP-Laptop erstellt und getestet. Dieser gehört dem Projektleiter und hat folgende Eckdaten:

HP EliteBook 840 G2	
Betriebssystem	Windows 7 Pro
Prozessor	Intel Core i7 5500U / 2.4 GHz
Arbeitsspeicher	8 GB DDR3L
Festplatte	256 GB SSD

11.2 Software

11.2.1 Dokumentation

Für die Dokumentation wird auf die Microsoft Produkte Word und Excel gesetzt. Für Uml-Diagramme wird die kostenlose Software AgroUml verwendet.

11.2.2 Softwareentwicklung

Für die Softwareentwicklung wird die SpringSource Toolsuite eingesetzt. Diese beinhaltet neben diversen kleinen Plugins vor allem Spring Roo welches das Erstellen einer Webapplikation erleichtert.

11.3 Daten

Zur Versionsverwaltung von Source-Code und Dokumentation wird Git eingesetzt. Dazu wurde auf github.com ein entsprechendes Repository eröffnet.

Die Daten werden somit auf dem oben erwähnten Laptop und auf den Servern von github.com gesichert.

12 Änderungsmanagement

Änderungen welche nicht während der Anforderungsanalyse aufgenommen wurden, werden als Change-Request angesehen. Jeder Change-Request wird durch folgende Prozess bewertet. Damit ist sichergestellt, dass das Projekt den Zeitrahmen einhalten kann.

