HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

INSTITUT
FÜR
SOFTWARE

Mitarbeiter & Absenzen - JSF Showcase

Bachelorarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Frühjahrssemester 2012

Autoren:	Beat Bodenmann, Christian Zurbuchen
Betreuer:	Prof. Hans Rudin
Experte:	Daniel Hiltebrand, CREALOGIX E-Business AG
Institut:	Institut für Software

Aufgabenstellung

Aufgabenstellung Bachelorarbeit für Herrn Beat Bodenmann und Herrn Christian Zurbuchen „Mitarbeiter & Absenzen - JSF Showcase“

1. Auftraggeber, Betreuer und Experte

Bei dieser Arbeit handelt es sich um ein Thema, das von den Studierenden eingebracht wurde.

Betreuer HSR: Prof. Hans Rudin, HSR, hrudin@hsr.ch , +41 55 222 49 36
Experte: Daniel Hiltebrand, Crealogix

2. Studierende

Diese Arbeit wird als Bachelorarbeit an der Abteilung Informatik durchgeführt von

- Beat Bodenmann, bbodenma@hsr.ch
- Christian Zurbuchen, czurbuch@hsr.ch

3. Aufgabenstellung

Diese Arbeit hat zum Ziel, eine Web-Applikation zur Verwaltung von Mitarbeitern und Absenzen in einem Unternehmen zu entwickeln. Bei der Entwicklung soll das moderne Java Standard Web Framework JSF2 mit geeigneten weiteren Bibliotheken zum Einsatz kommen. Die Applikation soll zudem als Beispielapplikation im Modul Internettechnologien an der HSR dienen. Erwartet wird auch ein Erfahrungsbericht und eine Beurteilung der eingesetzten Technologien und Werkzeuge.

Nachfolgend eine grobe Beschreibung des Funktionsumfangs der geplanten Applikation (übernommen aus dem Themenvorschlag der Studierenden mit kleinen Anpassungen). Die genauen Anforderungen werden von den Studierenden im Rahmen dieser Arbeit ausgearbeitet.

Funktionsumfang

Die Applikation ist in zwei Bereiche unterteilt:

1. Mitarbeiter (Abteilungen, Projekte)
2. Absenzen

Mitarbeiter (Abteilungen, Projekte)

In der Applikation können einzelne Abteilungen und Projekte erfasst werden.

Mitarbeiter können mit Basisinformationen (Name, Telefonnummern, Mailadressen, usw.) erstellt werden. Ein Mitarbeiter kann anschliessend einer Abteilung und einem oder mehreren Projekten (unter Angabe eines Prozentsatzes) hinzugefügt werden. In einer Abteilung oder einem Projekt kann er als Abteilungsleiter bzw. als Projektleiter definiert werden.

Absenzen

Jeder Mitarbeiter kann seine Absenzen (Militär, Ferien, Kurse, usw.) über die Applikation erfassen. Ein Abteilungsleiter oder ein Projektleiter kann seinen Mitarbeitern bzw. Projektmitarbeitern gewisse Absenzen genehmigen. Ein normaler Mitarbeiter kann dadurch z.B. seine Ferien nur als Wunsch eingeben. Dadurch wird es einen Genehmigungsprozess für gewisse Absenzen geben.

Der Abteilungsleiter oder ein Projektleiter hat anschliessend die Möglichkeit, die Absenzen seiner Abteilung bzw. seines Projektes in einer Übersichtsseite zu überblicken, um zukünftige Planungen vorzunehmen. Auch soll bei Projekten ersichtlich sein, zu wie viel Prozent sich einzelne Mitarbeiter am Projekt beteiligen.

Ausbaumöglichkeiten

Folgende Ausbaumöglichkeiten bestehen:

- Implementierung von Mobile Clients mit spezifischem Funktionsumfang
- Export von Auswertungen und Statistiken
- Genehmigungsprozess konfigurierbar

4. Zur Durchführung

Mit dem Betreuer finden wöchentliche Besprechungen statt. Zusätzliche Besprechungen sind nach Bedarf durch die Studierenden zu veranlassen.

Alle Besprechungen sind von den Studierenden mit einer Traktandenliste vorzubereiten, die Besprechung ist durch die Studierenden zu leiten und die Ergebnisse sind in einem Protokoll festzuhalten, das den Betreuern und dem Auftraggeber per E-Mail zugestellt wird.

Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. An Meilensteinen gemäss Projektplan sind einzelne Arbeitsresultate in vorläufigen Versionen abzugeben. Über die abgegebenen Arbeitsresultate erhalten die Studierenden ein vorläufiges Feedback. Eine definitive Beurteilung erfolgt auf Grund der am Abgabetermin abgelieferten Dokumentation.

5. Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen (siehe https://www.hsr.ch/Allgemeine-Infos-Diplom-Bach.4418.0.html). Die zu erstellenden Dokumente sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollten den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Alle Resultate sind vollständig auf CD/DVD in 3 Exemplaren abzugeben. Der Bericht ist ausgedruckt in doppelter Ausführung abzugeben.

6. Termine

Siehe auch Terminplan auf https://www.hsr.ch/Termine-Diplom-Bachelor-und.5142.0.html.

Montag, den 20. Februar 2012	**Beginn der Bachelorarbeit,** **Ausgabe der Aufgabenstellung durch die Betreuer**
8. Juni 2012	Abgabe Kurzbeschreibung und A0-Poster. Vorlagen stehen unter den allgemeinen Infos Diplom-, Bachelor- und Studienarbeiten zur Verfügung.
15. Juni 2012, 12:00	**Abgabe** der Arbeit an den Betreuer bis 12.00 Uhr. Fertigstellung des A0-Posters bis 12.00 Uhr. Abgabe der Posters im Abteilungssekretariat 6.113.
15. Juni 2012	HSR-Forum, Vorträge und Präsentation der Bachelor- und Diplomarbeiten, 13 bis 18 Uhr
6.8. - 25.08.2012	Mündliche Prüfung zur Bachelorarbeit

7. Beurteilung

Eine erfolgreiche Bachelorarbeit zählt 12 ECTS-Punkte pro Studierenden. Für 1 ECTS-Punkt ist eine Arbeitsleistung von 30 Stunden budgetiert (Siehe auch Modulbeschreibung der Bachelorarbeit https://unterricht.hsr.ch/staticWeb/allModules/19419_M_BAI.html.)

Für die Beurteilung ist der HSR-Betreuer verantwortlich.

Gesichtspunkt	Gewicht
1. Organisation, Durchführung	1/6
2. Berichte (Abstract, Management Summary und andere) sowie Gliederung, Darstellung, Sprache der gesamten Dokumentation	1/6
3. Inhalt*)	3/6
4. Mündliche Prüfung zur Bachelorarbeit	1/6

*) Die Unterteilung und Gewichtung von 3. Inhalt wird im Laufe dieser Arbeit mit den Studierenden festgelegt.

Im Übrigen gelten die Bestimmungen der Abteilung Informatik für Bachelorarbeiten.

Rapperswil, den 19. Februar 2012

Hans Rudin

Prof. Hans Rudin
Institut für Software
Hochschule für Technik Rapperswil

Eigenständigkeitserklärung

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt sind oder mit dem Betreuer schriftlich vereinbart wurden,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben haben,
- dass wir keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt haben.

Rapperswil, 14. Juni 2012

Beat Bodenmann

Christian Zurbuchen

Inhaltsverzeichnis

Aufgabenstellung ... 2
Eigenständigkeitserklärung ... 5
Inhaltsverzeichnis ... 6
Abstract ... 13
Management Summary ... 14
1. Ausgangslage ... 14
2. Vorgehen / Technologien ... 15
3. Ergebnisse ... 16
3.1 Applikation ... 16
3.2 Showcase ... 17
Anforderungsspezifikation ... 18
1. Einführung ... 18
1.1 Zweck ... 18
1.2 Beschreibung ... 18
1.3 Referenzen ... 18
1.4 Übersicht ... 18
2. Allgemeine Beschreibung ... 19
2.1 Produkt Perspektive ... 19
2.2 Produkt Funktion ... 19
2.2.1 Mitarbeiter (Abteilungen, Projekte) ... 19
2.2.2 Absenzen ... 20
2.3 Benutzer Charakteristik ... 21
2.4 Einschränkungen ... 21
2.5 Annahmen ... 21
2.6 Abhängigkeiten ... 21
3. Use Cases ... 22
3.1 Use Case Diagramm ... 22
3.2 Aktoren ... 23
3.2.1 Mitarbeiter ... 23
3.2.2 Abteilungen & Projekte ... 23
3.2.3 Absenzen ... 23
3.3 Beschreibungen (Brief) ... 24
3.3.1 UC1: Login ... 24
3.3.2 UC2: Change password ... 24
3.3.3 UC3: CRUD employee ... 24
3.3.4 UC4: CRUD absence ... 24

3.3.5 UC5: Approve/Disapprove absence 25
3.3.6 UC6: CRUD department 25
3.3.7 UC7: CRUD project 25
3.3.8 UC8: Manage absence types 26
3.4 Beschreibungen (Fully Dressed) 26
4. Weitere Anforderungen 27
4.1 Qualitätsmerkmale 27
4.1.1 Funktionalität 27
4.1.2 Zuverlässigkeit 27
4.1.3 Benutzbarkeit 27
4.1.4 Effizienz 27
4.1.5 Wartbarkeit 27
4.1.6 Übertragbarkeit 27
4.2 Schnittstellen 28
4.2.1 Softwareschnittstelle 28
4.2.2 Datenbankschnittstelle 28
Domainanalyse 29
1. Einführung 29
1.1 Gültigkeitsbereich 29
1.2 Referenzen 29
1.3 Übersicht 29
2. Domain Modell 30
2.1 Strukturdiagramm 30
2.2 Wichtige Konzepte 31
2.2.1 Employee 31
2.2.2 Department 32
2.2.3 Project 33
2.2.4 Worker 33
2.2.5 Absence 34
2.2.6 AbsenceType 34
3. Systemsequenzdiagramme 35
3.1 UC1: Login 35
3.2 UC2: Change password 35
3.3 UC3: CRUD employee 35
3.4 UC4: CRUD absence 36
3.5 UC5: Approve/Disapprove absence 36
3.6 UC6: CRUD department 36
3.7 UC7: CRUD project 37

3.8 UC8: Manage absence types 37
4. Systemoperationen 38
4.1 Vertrag Systemoperation UC1: Login 38
4.2 Vertrag Systemoperation UC2: Change password 38
4.3 Vertrag Systemoperation UC3: CRUD employee 38
4.4 Vertrag Systemoperation UC4: CRUD absence 39
4.4.1 Variante 1 39
4.4.2 Variante 2 39
4.5 Vertrag Systemoperation UC5: Approve/Disapprove absence 39
4.6 Vertrag Systemoperation UC6: CRUD department 40
4.7 Vertrag Systemoperation UC7: CRUD project 40
4.8 Vertrag Systemoperation UC8: Manage absence types 40
Technischer Bericht 41
1. Evaluation Java EE Technologien 41
1.1 Applikationsserver 41
1.1.1 Typen 41
1.1.2 Gegenüberstellung 44
1.2 User Interface Komponente 45
2. Technologische Aspekte 46
2.1 JSF Template & Primefaces 46
2.1.1 Problem 46
2.1.2 Lösung 47
2.2 Doppelklick 48
2.2.1 Problem 48
2.2.2 Lösung 49
2.3 Schedule Komponente 50
2.3.1 Problem 50
2.3.2 Lösung 50
2.4 Entity Manager (Factory) 51
2.4.1 Problem 51
2.4.2 Lösung 51
3. Fachliche Aspekte 52
3.1 Rechte & Rollen 52
3.1.1 Problemstellung 52
3.1.2 Lösung 52
3.2 Konfigurierbarkeit 54
3.2.1 Problemstellung 54
3.2.2 Effektive Konfigurierbarkeit 54

Software Architektur 55
1. Einführung 55
1.1 Zweck 55
1.2 Gültigkeitsbereich 55
1.3 Referenzen 55
1.4 Übersicht 55
2. Systemübersicht 56
2.1.1 Client Tier 56
2.1.2 Web Tier 56
2.1.3 Business Tier 57
2.1.4 EIS Tier 57
3. Architektonische Ziele & Einschränkungen 58
3.1 Implementationsstrategie 58
3.1.1 Daten 58
3.1.2 Kommunikation 58
3.1.3 Sicherheit 58
3.1.4 Session Handling 59
3.2 Verwendete Technologien 60
4. Logische Architektur 61
4.1 User Interfaces 61
4.1.1 Schnittstellen 62
4.1.2 Wichtige interne Abläufe 62
4.2 Geschäftslogik 63
4.2.1 Subpackages 64
4.2.2 Schnittstellen 68
4.2.3 Wichtige interne Abläufe 68
4.3 Datenzugriffsschicht 69
4.3.1 Subpackages 70
4.3.2 Schnittstellen 73
4.4 Wichtige Abläufe 74
5. Prozesse und Threads 76
6. Deployment 76
7. Datenspeicherung 77
8. Grössen und Leistung 78
8.1 Mitarbeiter 78
8.2 Projekte 78
8.3 Absenzen 78
Handbuch (Inbetriebnahme / Betrieb) 79

1. Inbetriebnahme 79
1.1.1 Starten der Servern 79
1.1.2 Installation von teamMgmt 79
2. Betrieb 80
2.1 Server 80
2.1.1 Standardeinstellungen 80
2.1.2 Starten der Domäne 80
2.1.3 Stoppen der Domäne 80
2.1.4 Überprüfung laufender Domänen 81
2.1.5 Starten des Java Datenbank Servers 81
2.1.6 Stoppen des Java Datenbank Servers 81
2.1.7 Remote Administration aktivieren 81
2.1.8 Zugriff Administrationskonsole 81
Projektmanagement 82
1. Projektplan 82
1.1 Einführung 82
1.1.1 Zweck 82
1.1.2 Gültigkeitsbereich 82
1.1.3 Referenzen 82
1.1.4 Übersicht 82
1.2 Projekt Übersicht 83
1.2.1 Zweck und Ziel 83
1.2.2 Lieferumfang 83
1.3 Projektorganisation 84
1.3.1 Organisationsstruktur 84
1.3.2 Externe Schnittstellen 84
1.4 Management Abläufe 85
1.4.1 Kostenvoranschlag 85
1.4.2 Zeitliche Planung 85
1.4.3 Besprechungen 88
1.4.4 Releases 88
1.5 Risikomanagement 89
1.5.1 Risiken 89
1.5.2 Umgang mit Risiken 89
1.6 Arbeitspakete 89
1.7 Infrastruktur 90
1.8 Qualitätsmassnahmen 91
1.8.1 Dokumentation 91

1.8.2 Projektmanagement 92
1.8.3 Entwicklung 92
1.8.4 Testen 93
2. Technische Risiken 94
3. Iterationspläne und –assessments 95
4. Zeitauswertung 95
4.1 Disziplinen 95
4.2 Mitglieder 96
Qualitätssicherung 97
1. Test 97
1.1 Einführung 97
1.2 Systemtestspezifikation 97
1.2.1 Voraussetzungen 97
1.2.2 Systemtest 97
1.3 Testabdeckung 116
2. Code Analyse 117
2.1 Kennzahlen 117
2.2 Struktur 117
2.2.1 Package bll 117
2.2.2 Package dal 118
2.3 Metriken 119
2.3.1 Cyclomatic Complexity 119
2.3.2 Feature Envy 120
2.3.3 Lines of Code in Method 121
2.3.4 Efferent Coupling 122
2.3.5 Lack of Cohesion 123
2.4 FindBugs 123
Erfahrungen Technologien 124
1. GlassFish 124
2. EJB & JPA 124
3. JSF 125
4. Primefaces 125
Persönliche Berichte 126
1. Beat Bodenmann 126
1.1 Projektverlauf 126
1.2 Rückblick 126
1.3 Lessons Learned 126
2. Christian Zurbuchen 127

2.1 Projektverlauf 127
2.2 Rückblick 127
2.3 Lessons Learned 128
Anhang A 129
1. Glossar 129
2. Verzeichnisse 131
2.1 Literatur 131
2.2 Tabellen 132
2.3 Abbildungen 133
Anhang B 135

Abstract

Bei teamMgmt handelt es sich um eine Java EE Web Applikation, welche mit dem Framework JSF 2.0 implementiert wurde. Die dazugehörige Datenbank läuft auf einem GlassFish Applikationsserver und die Persistierung erfolgt mit EJB 3.1 über JPA 2.0. Mittels Primefaces 3.2 wurden die User Interfaces erstellt.

Die Applikation vereinfacht einem kleinen bis mittelgrossen Unternehmen die Verwaltung der Mitarbeiter und deren Absenzen. Sie bietet die Möglichkeit Mitarbeiter nach Abteilung und Projekten zu gliedern. Basierend auf verteilten Rollen auf den erfassten Mitarbeitern, können die Daten innerhalb der Applikation verwaltet und somit zum Beispiel auch Absenzen genehmigt oder abgelehnt werden.

Die antreibende Idee dahinter ist, dass es für einen Abteilungsleiter oder Projektleiter schwierig ist, die Kapazitäten seines Teams optimal einzusetzen. Absenzen der Mitarbeiter erschweren diese Arbeit zusätzlich. An diesem Punkt greift teamMgmt an. Eine Firmen- oder Teamstruktur kann aufgebaut werden, die darin erfassten Mitarbeiter können Absenzen eintragen und deren Vorgesetzten diese genehmigen oder ablehnen. Es wird eine kompakte Übersicht nach Organisationseinheit, Abteilung oder Projekt geboten, welche den Planungsprozess unterstützt. Die Applikation teamMgmt wurde bewusst schlank gehalten um eine schnelle und intuitive Bedienung zu ermöglichen.

Ausserdem stellt diese Applikation ein Showcase für das moderne Framework JSF 2.0 dar. Ziel war es die neusten Technologien in diesem Zusammenhang zu verwenden und auch einen entsprechenden Erfahrungsbericht darauf aufzubauen.

Management Summary

1. Ausgangslage

Das Ziel dieser Arbeit ist es einem kleinen bis mittelgrossen Unternehmen die Verwaltung der Mitarbeiter und deren Absenzen zu ermöglichen. Mittels der Software können Mitarbeiter mit Kontaktinformationen, Abteilungen und Projekte hinzugefügt werden. Jeder kann seine eigenen Absenzen eintragen und verwalten. Ein Abteilungsleiter oder Projektleiter hat zusätzlich die Möglichkeit, die Absenzen seines Teams zu genehmigen oder abzulehnen. Dies vereinfacht den Überblick, wer wann abwesend sein wird und ermöglicht eine einfachere Zeitplanung. Ausserdem soll diese Applikation die meisten Aspekte von Java EE abdecken und somit das Zusammenspiel dieser neuen Technologien aufzeigen.

Es handelt sich um eine webbasierte Applikation mit dem Namen teamMgmt entstanden. Diese Applikation wird auf einem GlassFish Server, welcher Open Source und somit frei erhältlich ist, installiert. Nach einer erfolgreichen Installation kann die Applikation von beliebig vielen Clients via Webbrowser genutzt werden. Dies bringt den Vorteil, dass die Applikation an nur einem Ort gewartet werden muss, birgt aber gleichzeitig das Risiko des Single Point of Failure (SPOF), was bedeutet, dass bei einem Ausfall des Servers die Applikation für alle Clients nicht mehr verfügbar wäre.

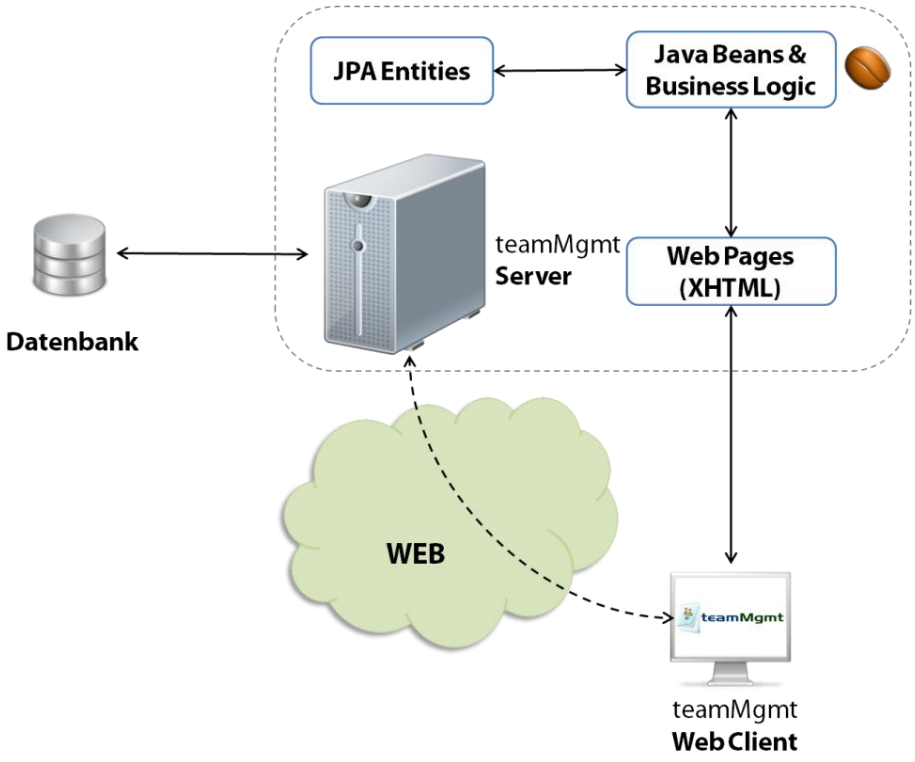

Abbildung 1: Management Summary: Systemarchitektur

2. Vorgehen / Technologien

Zu Beginn der Arbeit wurde eine Analyse der verfügbaren Technologien im Bereich der Applikationsserver durchgeführt. Es standen die Produkte JBoss, GlassFish, WebLogic und Websphere zur Diskussion, wobei es sich bei den letzten Beiden um kostenpflichtige Varianten handelt, welch somit bereits früh eliminiert wurden. Eine Auswahl zwischen JBoss und GlassFish zu treffen war nicht einfach, da beide praktisch dieselben Merkmale besitzen. Die Entscheidung fiel schlussendlich auf GlassFish, weil es sich bei GlassFish um die Referenzimplementation des Java EE Standards handelt.

Für die Entwicklung der User Interfaces wurden verschiedene Ansätze überprüft. Die grosse Herausforderung dabei war das Zusammenführen all dieser neuen Technologien innerhalb einer Applikation. Bei der Kombination des JSF 2.0 Frameworks mit Primefaces zur Erstellung von User Interfaces tauchten diverse Probleme auf. Gewisse Komponenten von JSF funktionierten mit Primefaces nicht ordnungsgemäss, und es mussten Alternativen gesucht werden.

Zudem stellte auch die Zuteilung der einzelnen Rollen und Rechte von Abteilungsleitern und Projektleitern eine weitere Schwierigkeit dar. Aufgrund der gegenseitigen Beziehungen der einzelnen Rollen musste einige Spezialfälle abgedeckt werden. Als Rollen werden die Funktionen der Mitarbeiter in der Firma beschreiben.

Um eine maximale Qualität des Codes und der Dokumentation zu erhalten wurden innerhalb des Projektteams die Verantwortlichkeiten klar aufgeteilt. Jedes Dokument wurde von einer anderen Person gegengelesen und der Code wurde individuell getestet.

3. Ergebnisse

3.1 Applikation

Die Applikation teamMgmt ermöglicht einem kleinen bis mittelgrossen Unternehmen die Verwaltung der Mitarbeiter und deren Absenzen. Es können Mitarbeiter mit Kontaktinformationen, Abteilungen und Projekte hinzugefügt werden. Jeder kann seine eigenen Absenzen eintragen und verwalten. Die Mitarbeiter besitzen ihren Position und Zuteilung entsprechende Rollen. Administrator, Abteilungsleiter, Projektleiter, Projektmanager und Mitarbeiter sind die verfügbaren Rollen.

Bei einem Administrator handelt es sich um einen Benutzer, welcher grundsätzlich alle Rechte besitzt. Ein beliebiger Mitarbeiter kann zu einem Administrator ernannt werden.

Ein Mitarbeiter wird zum Abteilungsleiter, sobald er bei einer Abteilung als solcher eingetragen wird. Er besitzt danach die Rechte alle Mitarbeiter dieser Abteilung und allfälligen Unterabteilungen zu verwalten und deren Absenzen zu genehmigen oder abzulehnen. Ausserdem kann er neue Projekte erstellen und dabei dieselben Mitarbeiter als Projektleiter oder Projektmanager eintragen.

Projektleiter sind jene Mitarbeiter, welche in einem Projekt als Projektleiter eingetragen wurden. Ein Projektleiter kann die Absenzen der Mitarbeiter seiner Projekte genehmigen, sofern diese nicht länger als einen Tag dauern und es sich nicht um periodisch wiederkehrende Absenzen handelt. Ein Projektmanager hat zusätzlich noch die Möglichkeit den Projektleiter seiner Projekte festzulegen. Die Position des Projektmanagers kann nach dem Erstellen lediglich von einem Administrator angepasst werden.

Mittels der kompakten Übersicht über die erfassten Absenzen (nur Genehmigte) erhält man je nach Wunsch eine Darstellung für einzelne Abteilungen oder Projekte. Diese kann in Monats-, Wochen- oder Tagesansicht angezeigt werden. Diese Übersicht wird uneingeschränkt jedem Mitarbeiter zur Verfügung gestellt, damit jeder seine zukünftigen Absenzen sinngemäss planen kann.

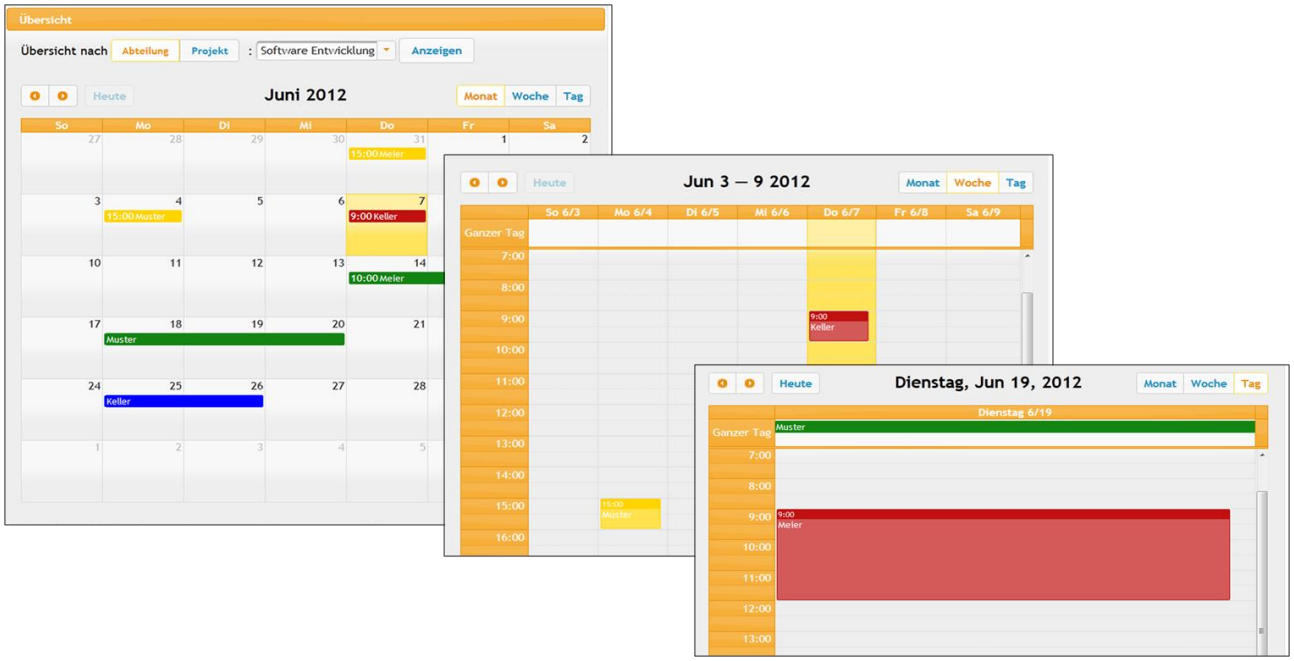

Abbildung 2: Management Summary: Übersicht der Absenzen

3.2 Showcase

Die Applikation arbeitet mit dem Framework Standard Java Server Faces 2.0 zur Entwicklung von grafischen Benutzeroberflächen für Web Applikationen. Dieses Framework wird durch das Komponentenframework Primefaces unterstützt. Die Geschäftslogik wurde mit Java Beans und Enterprise Java Beans umgesetzt. Letztere werden als Entities verwendet, welche mit Hilfe der Java Persistence API in einer Java Datenbank auf einem modernen GlassFish Applikationsserver gespeichert werden. Das Zusammenspiel all dieser neuen Technologien wird in der Applikation aufgezeigt und somit kann teamMgmt als Showcase für moderne Webapplikationen genutzt werden.

Anforderungsspezifikation

1. Einführung

1.1 Zweck

Dieses Dokument beschreibt die Anforderungen an das Produkt teamMgmt.

1.2 Beschreibung

Das Ziel dieser Arbeit ist es einem kleinen bis mittelgrossen Unternehmen die Verwaltung der Mitarbeiter und deren Absenzen zu erleichtern. Ausserdem soll es möglich sein, diverse spezifische Auswertungen dieser Absenzen zu generieren, um sich den benötigten Überblick verschaffen zu können. Von den einzelnen Mitarbeitern werden Kontaktinformationen (Name, Telefonnummern, Mailadressen, usw.) hinterlegt, um diese kontaktieren zu können.

Die antreibende Idee dahinter ist, dass es bis heute noch viele Unternehmen gibt, welche diese Arbeit mittels einer zentral abgelegten Excel Tabelle oder Ähnlichem lösen. Diese bieten meist keinen kompakten Überblick zum Beispiel für einen Projektleiter, da er sich die einzelnen Mitarbeiter seines Teams über die gesamte Tabelle zusammensuchen muss. Solche und auch noch weitere Probleme werden durch teamMgmt behoben.

Die Lösung wird in Form einer Web Applikation geboten, welche das moderne Java Standard Web Framework JSF 2.0 verwendet.

1.3 Referenzen

Nachfolgend werden die Dokumente aufgeführt, welche von diesem Dokument referenziert werden.

- Aufgabenstellung.pdf (Betreuer)
- Norm ISO 9126 (http://de.wikipedia.org/wiki/ISO/IEC_9126) (Stand 10.03.2012
- Glossar.pdf

1.4 Übersicht

Im folgenden Kapitel *Allgemeine Beschreibung* wird eine allgemeine Beschreibung des Produktes teamMgmt dargelegt. Die Hauptfunktionen der Applikation werden aus Sicht des Benutzers formuliert.

Sämtliche Use Cases werden im Format „brief“ und die Wichtigsten im Format „fully dressed“ im Kapitel *Use Cases* beschrieben.

Im letzten Kapitel *Weitere Anforderungen* werden Qualitätsmerkmale, Schnittstellen und Randbedingungen der Software erläutert.

2. Allgemeine Beschreibung

2.1 Produkt Perspektive

Die Applikation ermöglicht es, jedem Mitarbeiter einer Unternehmung Kontaktdaten und seine Absenzen zu erfassen. Einem Abteilungsleiter oder Projektleiter/-manager wird somit ein Überblick über die Absenzen seiner Abteilung bzw. seines Projektes gewährt.

Da unser Produkt für eine Vielzahl von unterschiedlichsten Unternehmungen interessant sein kann, wird im Verlauf des Projektes noch überprüft, ob auch eine Version für Mobile Clients entwickelt werden soll.

2.2 Produkt Funktion

In diesem Kapitel werden die Funktionen und somit die Anforderungen an das Produkt teamMgmt definiert. Die Funktionen werden hier nur beschrieben und aufgelistet. Später im Kapitel *Use Cases* werden diese detaillierter beschrieben und gruppiert.

Grundsätzlich ist die Applikation ist in zwei Bereiche unterteilt:

- Mitarbeiter (Abteilungen, Projekte)
- Absenzen

Auf diese Bereiche und deren spezifischen Anforderungen wird in den folgenden Kapiteln eingegangen. Allgemeine Anforderungen an die Applikation sind in der folgenden Tabelle beschrieben.

Nr	Funktion / Anforderung / Einschränkungen
1	Pro Mitarbeiter gibt es einen Account (inkl. Benutzername und Passwort).
2	Beim Aufruf der Applikation muss sich der Benutzer mittels Benutzername und Passwort authentifizieren.
3	Ein Mitarbeiter hat die Möglichkeit sein Passwort anzupassen.
4	Ein Mitarbeiter kann sich nur einmal gleichzeitig am System anmelden.

Tabelle 1: Anforderungsspezifikation: Allgemeine Anforderungen

2.2.1 Mitarbeiter (Abteilungen, Projekte)

In der Applikation können einzelne Abteilungen und Projekte erfasst werden. Mitarbeiter können mit Kontaktdaten(Vorname, Name, Adresse, private und mobile Telefonnummer und E-Mail Adresse), einem Pensum und der Information ob sie Administrator sind oder nicht erstellt werden. Ein Mitarbeiter kann ebenfalls einer Abteilung und einem oder mehreren Projekten. In einer Abteilung oder einem Projekt kann er als Abteilungsleiter bzw. als Projektleiter/-manager definiert werden.

In der folgenden Tabelle sind die Anforderungen an den Bereich Mitarbeiter beschrieben.

Nr	Funktion / Anforderung / Einschränkungen
5	Ein Mitarbeiter kann seine eigenen Kontaktdaten bearbeiten. Diese Kontaktdaten beinhalten Name, Vorname, Adresse, private und mobile Telefonnummer und E-Mail Adresse.
6	Ein Abteilungsleiter kann alle Kontaktdaten aller Mitarbeiter unterhalb seiner Stufe bearbeiten. Zudem kann er das Pensum und die Abteilung anpassen.
7	Ein Administrator kann uneingeschränkt alle Kontaktdaten aller Mitarbeiter bearbeiten. Zudem kann er das Pensum, die Abteilung und das Administratorrecht anpassen.
8	Ein Administrator kann neue Mitarbeiter erstellen (mit Benutzername und Passwort).
9	Alle Benutzer können alle Kontaktdaten aller Mitarbeiter ansehen.
10	Alle Benutzer können nach allen Mitarbeitern suchen innerhalb der Applikation.
11	Nur ein Administrator kann Mitarbeiter löschen.
12	Ein Abteilungsleiter kann Abteilungen neu erfassen, bearbeiten oder löschen unterhalb seiner Stufe.

Nr	Funktion / Anforderung / Einschränkungen
13	Ein Administrator kann uneingeschränkt Abteilungen neu erfassen, bearbeiten oder löschen.
14	Ein Administrator oder Abteilungsleiter kann neue Projekte erstellen oder bestehende löschen.
15	Ein Projektleiter kann seine eigenen Projekte bearbeiten, jedoch nicht Projektleiter oder -manager anpassen.
16	Ein Projektmanager kann eigene Projekte bearbeiten, jedoch nicht Projektmanager anpassen.
17	Ein Administrator kann uneingeschränkt bestehende Projekte bearbeiten.
18	Neu erstellten Projekten muss immer ein Projektmanager zugeteilt werden.
19	Ein Administrator, Abteilungsleiter oder zugeteilter Projektleiter oder -manager hat die Möglichkeit Mitarbeiter des Projektes hinzuzufügen oder zu entfernen. Dies mit einem Beginn- und Enddatum der Dauer des Arbeitseinsatzes.

Tabelle 2: Anforderungsspezifikation: Anforderungen Bereich Mitarbeiter (Abteilungen, Projekte)

2.2.2 Absenzen

Jeder Mitarbeiter kann seine Absenzen (Militär, Ferien, Kurse, usw.) über die Applikation erfassen. Ein Abteilungsleiter oder ein Projektleiter/-manager kann seinen Mitarbeitern bzw. Projektmitarbeitern Absenzen genehmigen. Ein Projektleiter/-manager kann dies nur, sofern die Absenz die Dauer eines Arbeitstages nicht überschreitet und es sich nicht um eine periodisch wiederkehrende Absenz handelt. Ein normaler Mitarbeiter gibt seine Absenz als offen ein und diese kann nun von der entsprechenden leitenden Person genehmigt oder abgelehnt werden. Der Abteilungsleiter oder ein Projektleiter/-manager hat anschliessend die Möglichkeit, die Absenzen seiner Abteilung bzw. seines Projektes in einer Übersichtsseite zu überblicken, um zukünftige Planungen vorzunehmen.

In der folgenden Tabelle sind die Anforderungen an den Bereich Absenzen beschrieben.

Nr	Funktion / Anforderung / Einschränkungen
20	Ein Mitarbeiter kann eigene Absenzen neu erstellen, diese mit offenem Status. Bearbeiten kann er nur offene Absenzen (keine Genehmigte oder Abgelehnte) und löschen kann er alle. Er hat ausserdem die Möglichkeit abgelehnte Absenzen wieder in den offenen Status zu bringen.
21	Ein Abteilungsleiter kann Absenzen aller Mitarbeiter unterhalb seiner Stufe genehmigen oder ablehnen.
22	Ein Projektleiter/-manager kann Absenzen aller Mitarbeiter innerhalb seiner Projekte genehmigen oder ablehnen, sofern diese die Dauer eines Tages nicht überschreiten oder es sich nicht um periodisch wiederkehrende Absenzen handelt.
23	Ein Administrator kann uneingeschränkt Absenzen aller Mitarbeiter genehmigen oder ablehnen.
24	Ein Administrator verwaltet die Absenztypen (wie Militär, Ferien, usw.) und kann uneingeschränkt Neue erstellen und Bestehende bearbeiten. Das Löschen eines Absenztypen hat das Löschen aller zugehörigen Absenzen zur Folge.
25	Jeder Mitarbeiter kann Absenzen aller Mitarbeiter geordnet nach Abteilungen oder Projekten in einer Übersichtsseite ansehen.

Tabelle 3: Anforderungsspezifikation: Anforderungen Bereich Absenzen

2.3 Benutzer Charakteristik

Das Produkt teamMgmt ist grundsätzlich für Unternehmungen unterschiedlichster Art und Grösse gedacht. Solange noch keine Arten von Mobile Clients geplant sind, ist die Applikation eher für Unternehmungen mit Büroräumlichkeiten geeignet, da z.B. auf der Baustelle nicht jeder Zugriff auf einen Computer hat. Die Applikation ist sehr leicht bedienbar, es wird lediglich vorausgesetzt, dass ein Benutzer einen Computer inklusive Webbrowser bedienen kann.

2.4 Einschränkungen

Das Benutzen eines Accounts zur Erfassung von Daten ist nur möglich, wenn für den gleichen Account keine andere Sitzung aktiv ist. Allfällige Konsistenzprobleme (gegenseitiges Überschreiben, Lost-Update) werden so ausgeschlossen. Ein Account kann somit maximal einmal zur gleichen Zeit am System angemeldet sein.

2.5 Annahmen

Es wird angenommen, dass die Benutzer über einen Computer inklusive Webbrowser und Netzwerkverbindung zum Server verfügen. Ausserdem muss von der Unternehmung, welche das Produkt verwendet, ein Server mit dem Service der Applikation verfügbar sein.

Bei einer möglichen Implementierung von Mobile Clients, wird zusätzlich angenommen, dass die Benutzer eines Mobile Clients die Verbindungsgebühren für die Datenübertragung zu einem Server in Kauf nehmen. Da moderne Smartphones wenn immer möglich Daten über WLAN beziehen und überdies die Option zur Verfügungen stellen die mobile Datenverbindung auszuschalten, stellt diese Tatsache kein Problem dar.

2.6 Abhängigkeiten

Die Applikation setzt ein betriebsbereites Deployment Package auf dem Server voraus, welches alles was für die Applikation nötig ist beinhaltet. Ausserdem sind eine Netzwerkverbindung zum Server und ein installierter Webbrowser beim Benutzer notwendig. Eine mobile Version der Applikation setzt ein Smartphone mit Internetzugang voraus. Beide Versionen sind nur funktionsfähig wenn eine Verbindung zum Server besteht.

3. Use Cases

3.1 Use Case Diagramm

Die beiden Teilbereiche innerhalb des Use Case Diagramms beziehen sich auf dasselbe Produkt. Übersichtshalber wurden sie aufgeteilt, damit nicht zu viele verwirrende Beziehungen das Diagramm unlesbar machen.

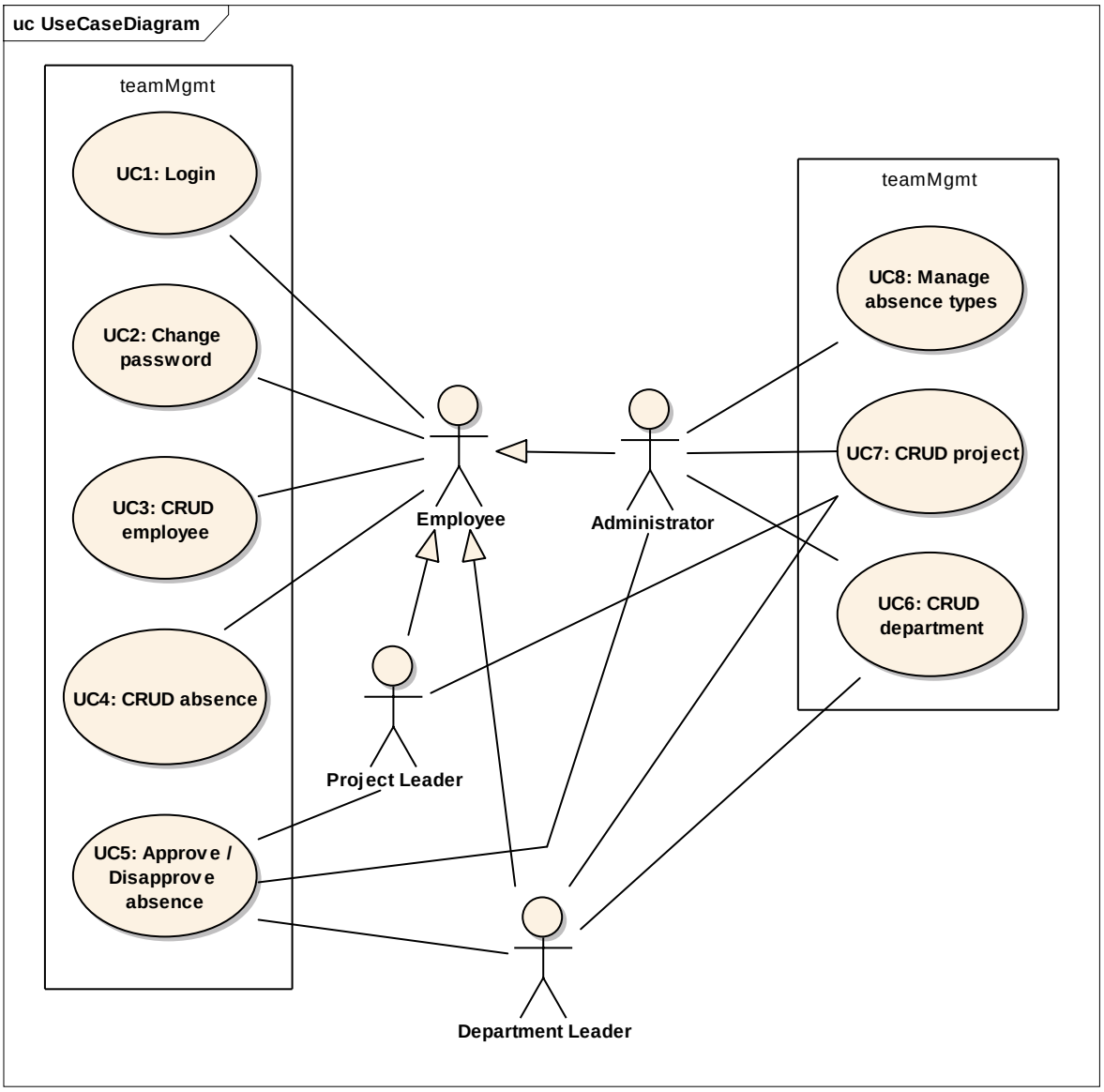

Abbildung 3: Anforderungsspezifikation: Use Case Diagramm

3.2 Aktoren

Die Applikation teamMgmt beinhaltet die folgenden Aktoren (Rollen):

- Employee
- Department Leader
- Project Leader (/ - Manager)
- Administrator

Jeder dieser Rollen verfügt über spezifische Rechte. Diese sind in den folgenden Kapiteln geordnet nach Bereichen genauer beschrieben. Die Titelzeile ist jeweils in folgende vier Bereiche unterteilt: C (Create, Erstellen), R (Read, Lesen), U (Update, Bearbeiten) und D (Delete, Löschen). Bei den Absenzen wurde diese ergänzt um M(Manage, Verwalten), was für eine Genehmigung oder Ablehnung von Absenzen steht.

3.2.1 Mitarbeiter

Die folgende Tabelle definiert die Rechte der einzelnen Rollen im Zusammenhang mit Mitarbeitern und deren Kontaktdaten. Es wurde unterteilt in die vier Bereiche eigene, unterhalb eigener Stufe, in seinem Projekt und alle. Dabei handelt es sich immer um Mitarbeiter und deren Kontaktdaten.

	Eigene				**Unterhalb eigener Stufe**				**In seinem Projekt**				**Alle**			
	C	**R**	**U**	**D**	**C**	**R**	**U**	**D**	**C**	**R**	**U**	**D**	**C**	**R**	**U**	**D**
Employee	X	X	X	X		X				X				X		
Department Leader	X	X	X	X	X	X	X	X		X				X		
Project Leader / - Manager	X	X	X	X		X				X				X		
Administrator	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Tabelle 4: Anforderungsspezifikation: Rollenrechte Bereich Mitarbeiter

3.2.2 Abteilungen & Projekte

Die folgende Tabelle definiert die Rechte der einzelnen Rollen im Zusammenhang mit Abteilungen und Projekten.

	Abteilungen				**Projekte**			
	C	**R**	**U**	**D**	**C**	**R**	**U**	**D**
Employee		X				X		
Department Leader		X	X		X	X		
Project Leader / - Manager		X				X	X	
Administrator	X	X	X	X	X	X	X	X

Tabelle 5: Anforderungsspezifikation: Rollenrechte Bereich Abteilungen & Projekte

3.2.3 Absenzen

Die folgende Tabelle definiert die Rechte der einzelnen Rollen im Zusammenhang mit Absenzen. Es wurde unterteilt in die vier Bereiche eigene, unterhalb eigener Stufe, in seinem Projekt und alle. Dabei handelt es sich immer um Absenzen der Mitarbeiter.

	Eigene					**Unterhalb eigener Stufe**					**In seinem Projekt**					**Alle**				
	C	**R**	**U**	**D**	**M**	**C**	**R**	**U**	**D**	**M**	**C**	**R**	**U**	**D**	**M**	**C**	**R**	**U**	**D**	**M**
Employee	X	X	X	X			X					X					X			
Department Leader	X	X	X	X	X		X			X		X					X			
Project Leader / - Manager	X	X	X	X	X		X			X		X			X		X			
Administrator	X	X	X	X	X		X			X		X			X		X			X

Tabelle 6: Anforderungsspezifikation: Rollenrechte Bereich Absenzen

3.3 Beschreibungen (Brief)

Auf die Rechte der einzelnen Rollen (Aktoren) wird in den Beschreibungen der Use Cases nicht mehr eingegangen, da diese bereits im Kapitel *Aktoren* genauer beschrieben sind.

3.3.1 UC1: Login

Use Case Name	UC1: Login
Aktoren	Employee
Beschreibung	Der Mitarbeiter (Mitarbeiter) kann sich mittels seines Benutzernamens und eines Passwortes in die Applikation einloggen.

Tabelle 7: Anforderungsspezifikation: UC1: Login

3.3.2 UC2: Change password

Use Case Name	UC2: Change password
Aktoren	Employee
Beschreibung	Der Mitarbeiter (Employee) kann sein Passwort nach erfolgreicher Eingabe des alten Passwortes problemlos ändern. Eine doppelte Eingabe des neuen Passwortes ist aus Sicherheitsgründen jedoch notwendig.

Tabelle 8: Anforderungsspezifikation: UC2: Change password

3.3.3 UC3: CRUD employee

Use Case Name	UC3: CRUD employee
Aktoren	Employee
Beschreibung	Beinhaltet das Erstellen, Lesen, Bearbeiten und Löschen von Mitarbeitern und deren Kontaktdaten. Abteilungsleiter können dies zusätzlich unterhalb ihrer Stufe. Administratoren können dies uneingeschränkt.

Tabelle 9: Anforderungsspezifikation: UC3: CRUD employee

3.3.4 UC4: CRUD absence

Use Case Name	UC4: CRUD absence
Aktoren	Employee
Beschreibung	Beinhaltet das Erstellen, Lesen, Bearbeiten und Löschen von Absenzen der einzelnen Mitarbeiter.

Tabelle 10: Anforderungsspezifikation: UC4: CRUD absence

3.3.5 UC5: Approve/Disapprove absence

Use Case Name	UC5: Approve/Disapprove absence
Aktoren	Administrator, Department Leader, Project Leader / - Manager
Beschreibung	– Ein Administrator kann ohne Einschränkung alle Absenzen aller Mitarbeiter genehmigen oder ablehnen. – Ein Abteilungsleiter (Department Leader) kann Absenzen eines Mitarbeiters innerhalb von Abteilungen unterhalb seiner Stufe genehmigen oder ablehnen. – Ein Projektleiter/-manager (Project Leader / - Manager) kann Absenzen, welche die Dauer eines Tages nicht überschreiten und nicht periodisch wiederkehrend sind von Mitarbeitern innerhalb seines Projektes genehmigen oder ablehnen.

Tabelle 11: Anforderungsspezifikation: UC5: Approve/Disapprove absence

3.3.6 UC6: CRUD department

Use Case Name	UC6: CRUD department
Aktoren	Administrator, Department Leader
Beschreibung	– Ein Administrator kann uneingeschränkt Abteilungen (Department) erstellen, bearbeiten und löschen. Die einzelnen Mitarbeiter können dann diesen Abteilungen zugeteilt werden oder als Abteilungsleiter gesetzt werden. – Ein Abteilungsleiter (Department Leader) kann Abteilungen unterhalb seiner Stufe erstellen, bearbeiten und löschen.

Tabelle 12: Anforderungsspezifikation: UC6: CRUD department

3.3.7 UC7: CRUD project

Use Case Name	UC7: CRUD project
Aktoren	Administrator, Department Leader, Project Leader / - Manager
Beschreibung	– Ein Administrator kann Projekte erstellen, bearbeiten und sogar löschen. Er kann ohne Einschränkungen alle Mitarbeiter einem Projekt zuteilen oder auch wieder entfernen und Anpassungen an Projektleiter und -manager vornehmen. – Ein Abteilungsleiter (Department Leader) kann Projekte erstellen. – Ein Projektleiter (Project Leader) kann Mitarbeiter zu seinen Projekt hinzufügen, jedoch nicht den Projektleiter oder -manager anpassen. – Ein Projektmanager (Project Manager) kann zusätzlich zum Projektleiter auch den Projektleiter anpassen.

Tabelle 13: Anforderungsspezifikation: UC7: CRUD project

3.3.8 UC8: Manage absence types

Use Case Name	UC8: Manage absence types
Aktoren	Administrator
Beschreibung	Nur ein Administrator kann Absenztypen (z.B. Militär, Ferien, oder Ähnliche) uneingeschränkt neu erstellen, bearbeiten oder löschen.

Tabelle 14: Anforderungsspezifikation: UC8: Manage absence types

3.4 Beschreibungen (Fully Dressed)

Die Beschreibung eines Use Cases im Format „fully dressed“ wäre im Falle eines CRUD Use Cases viel zu komplex und auf Grund der verschiedenen Rollen und deren Rechte nicht umsetzbar. Die anderen Use Cases sind für eine Beschreibung in diesem Format viel zu einfach, weshalb dies für die Entwicklung nicht von grossem Nutzen ist. Deshalb wurde auf Beschreibungen im Format „fully dressed“ verzichtet.

4. Weitere Anforderungen

4.1 Qualitätsmerkmale

Die Qualitätsmerkmale wurden an Hand der Norm ISO 9126 [1] beschrieben. Die wichtigsten Merkmale für das Produkt teamMgmt werden in den folgenden Abschnitten beschrieben.

4.1.1 Funktionalität

Um Zugriff auf die Applikation zu erhalten muss sich der Benutzer mittels eines Benutzernamens und eines persönlichen Passwortes einloggen. So wird unberechtigter Zugriff ausgeschlossen. Es liegt in der Verantwortlichkeit des Administrators der Software die Rechte korrekt zu verteilen um einen internen unberechtigten oder unnötigen Zugriff zu vermeiden.

Falls die mögliche Erweiterung mit Auswertungen in Excel oder PDF implementiert wird, so wird darauf geachtet, dass die exportierten Daten in den jeweiligen Applikationen weiterverwendet werden können.

4.1.2 Zuverlässigkeit

Bei fehlerhaften Eingaben wird die Applikation nicht direkt abstürzen, sondern entsprechende Fehlermeldungen anzeigen. Abstürze durch Systemfehler oder Ähnliches werden dem Benutzer verständlich mitgeteilt und zusätzlich wird dem Benutzer mitgeteilt, wie er weiter vorzugehen hat.

4.1.3 Benutzbarkeit

Die Applikation teamMgmt wird leicht und intuitiv zu bedienen sein. Ein Anwender kann ohne grossen Aufwand die Applikation und deren Funktionalitäten erlernen.

Zudem wird die Erfassung einer Absenz schnell und ohne Umwege möglich sein. Die Interaktionen zwischen System und Benutzer werden auf ein Minimum reduziert, es werden Standardwerte, wie beispielsweise das aktuelle Datum, vorgeschlagen.

4.1.4 Effizienz

Die Applikation wird kurze Antwort- und Verarbeitungszeiten haben. Auch wenn eine hohe Anzahl Benutzer das Produkt verwendet wird dies gewährleistet.

4.1.5 Wartbarkeit

Da die Applikation mit dem Test Driven Development Ansatz implementiert wird, wird eine mögliche Weiterentwicklung erleichtert und die einfache Testbarkeit ist somit garantiert. Der Code wird nach den geltenden Normen entwickelt. Ausserdem wird die Qualität des Codes regelmässig überprüft und verbessert und mittels Tools (FindBugs, STAN) werden Unschönheiten ausgemerzt. So kann ein einfach lesbarer und verständlicher Code garantiert werden, was wiederum die Wartbarkeit erheblich erhöht.

4.1.6 Übertragbarkeit

Die Software benötigt keine Installation, ausser ein vorinstalliertes JRE ist notwendig und das Vorhandensein einer Netzwerkverbindung zum Server und einem aktuellen Webbrowser. Somit kann die Software unabhängig von Betriebssystem oder Webbrowserart verwendet werden.

[1] Qualitätsmerkmale der Norm ISO 9126 (http://de.wikipedia.org/wiki/ISO/IEC_9126)

4.2 Schnittstellen

4.2.1 Softwareschnittstelle

Der Service läuft auf einem zentralen Server. Zur Benutzung des Services muss sich der Benutzer beim Server authentifizieren. Alle Benutzer müssen diesen Service verwenden um mit dem Server kommunizieren zu können.

4.2.2 Datenbankschnittstelle

Die Benutzer speichern lokal prinzipiell keine Daten (ausgenommen das Caching des Webbrowsers). Die Daten werden immer direkt an den Server gesendet. Dieser arbeitet mit einer Datenbank (integrierte Java Datenbank; Derby) und speichert die Daten direkt in der Datenbank ab.

Domainanalyse

1. Einführung

1.1 Gültigkeitsbereich

Die Gültigkeit dieses Dokuments erstreckt über das gesamte Projekt teamMgmt.

1.2 Referenzen

Nachfolgend werden die Dokumente aufgeführt, welche von diesem Dokument referenziert werden.

- Aufgabenstellung.pdf
- Projektplan.pdf
- Anforderungsspezifikation.pdf
- Glossar.pdf

1.3 Übersicht

Im Kapitel *Domain Modell* wird das Domainmodell des Produktes teamMgmt beschrieben und deren einzelne Komponenten werden ausführlich erklärt.

Die Sequenzdiagramme der kritischen und wichtigen Use Cases werden im Kapitel *Systemsequenzdiagramme* gezeigt.

Schlussendlich werden im letzten Kapitel *Systemoperationen* noch die wichtigsten Systemoperationen erklärt und Operation Contracts werden diesbezüglich definiert.

2. Domain Modell

2.1 Strukturdiagramm

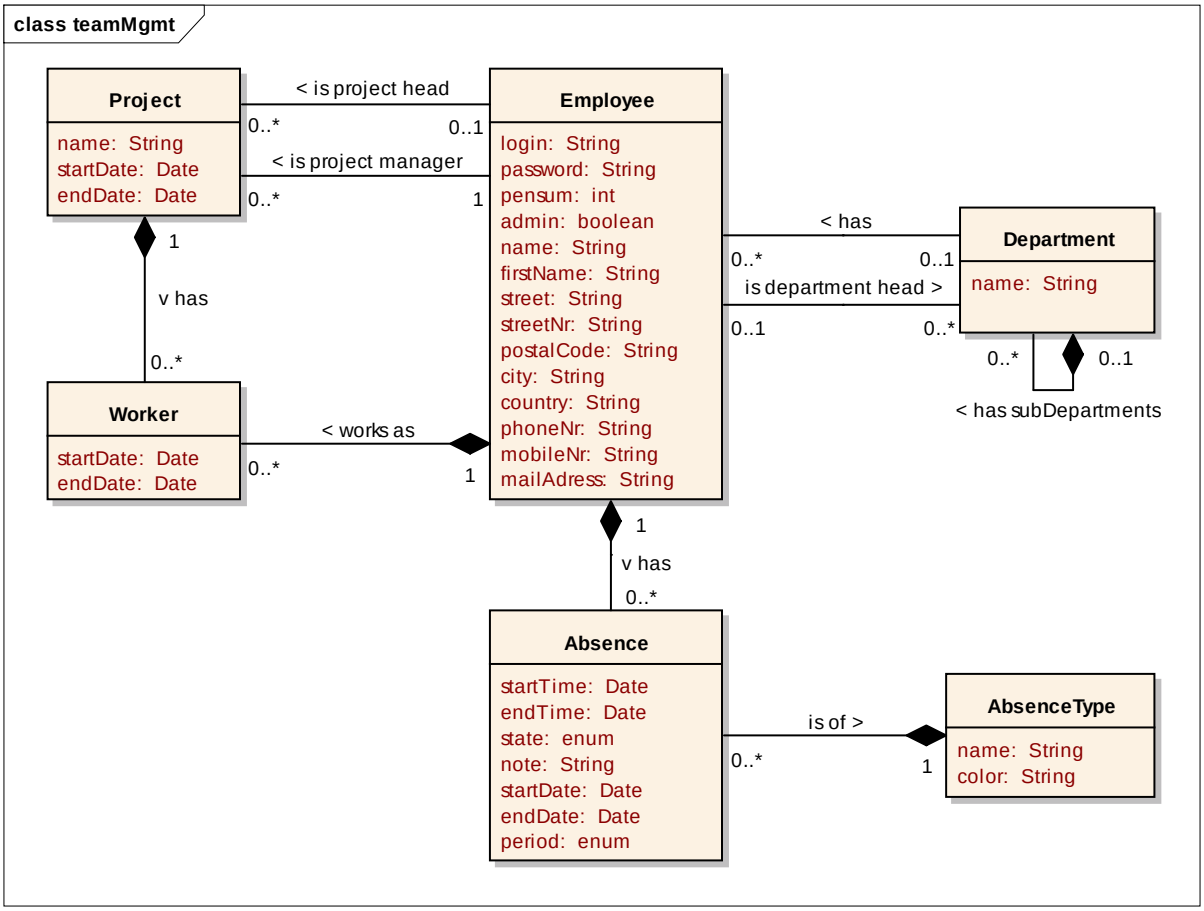

Abbildung 4: Domainanalyse: Domainmodell

2.2 Wichtige Konzepte

2.2.1 Employee

Das Konzept Employee beschreibt einen einzelnen Mitarbeiter des verwalteten Unternehmens. Dieser ist einer Abteilung (Department) zugeordnet und kann in mehreren Projekten (Project) tätig sein. Er kann mehrere Absenzen (Absence) haben und besitzt Kontaktdaten. Wird der Employee gelöscht so werden auch seine Absenzen, Kontaktdaten und die Mitarbeit in Projekten (Worker) gelöscht.

2.2.1.1 Attribute

Attribut	Beschreibung	Typ
login	Benutzername des Mitarbeiters	String
password	Passwort des Mitarbeiters	String
pensum	Arbeitspensum eines Mitarbeiters in Prozent	int
admin	Gibt an ob Administrator oder nicht	boolean
name	Name	String
firstName	Vorname	String
street	Strasse	String
streetNr	Strassennummer	String
postalCode	PLZ	String
city	Stadt oder Ort	String
country	Land	String
phoneNr	Telefonnummer (privat)	String
mobileNr	Telefonnummer (mobil)	String
mailAdress	E-Mail Adresse	String

Tabelle 15: Domainanalyse: Employee Attribute

2.2.1.2 Beziehungen

Multiplizität	zu Konzept	Beschreibung
0..* : 0..1	Department	Ein Mitarbeiter kann einer Abteilung zugeteilt sein.
0..1 : 0..*	Department	Ein Mitarbeiter kann von beliebig vielen Abteilungen Abteilungsleiter sein.
0..1 : 0..*	Project	Ein Mitarbeiter kann von beliebig vielen Projekten Projektleiter sein.
1 : 0..*	Project	Ein Mitarbeiter kann von beliebig vielen Projekten Projektmanager sein.
1 : 0..*	Worker	Ein Mitarbeiter kann in beliebig vielen Projekten tätig sein. Komposition auf Seite Mitarbeiter; Das Löschen eines Mitarbeiters hat auch die Löschung der zugehörigen Mitarbeit in Projekten zur Folge.
1 : 0..*	Absence	Ein Mitarbeiter kann beliebig viele Absenzen haben. Komposition auf Seite Mitarbeiter; Das Löschen eines Mitarbeiters hat auch die Löschung der zugehörigen Absenzen zur Folge.

Tabelle 16: Domainanalyse: Employee Beziehungen

2.2.2 Department

Das Konzept Department beschreibt die einzelne Abteilung der Unternehmung. Ein Department beinhaltet mehrere Mitarbeiter (Employee), wovon einer als Abteilungsleiter gekennzeichnet ist.

2.2.2.1 Attribute

Attribut	Beschreibung	Typ
name	Bezeichnung der Abteilung	String

Tabelle 17: Domainanalyse: Department Attribute

2.2.2.2 Beziehungen

Multiplizität	zu Konzept	Beschreibung
0..1 : 0..*	Employee	Eine Abteilung beinhaltet beliebig viele Mitarbeiter.
0..* : 0..1	Employee	Eine Abteilung besitzt einen oder keinen Abteilungsleiter.
0..1 : 0..*	Department	Eine Abteilung kann mehrere Unterabteilungen besitzen. Komposition bei der Abteilung; Eine Löschung einer Abteilung hat die Löschung möglicher Unterabteilungen zur Folge.

Tabelle 18: Domainanalyse: Department Beziehungen

2.2.3 Project

Das Konzept Project stellt ein Projekt innerhalb der Unternehmung dar.

2.2.3.1 Attribute

Attribut	Beschreibung	Typ
name	Projektname	String
startDate	Projektstart	Date
endDate	Projektende	Date

Tabelle 19: Domainanalyse: Project Attribute

2.2.3.2 Beziehungen

Multiplizität	zu Konzept	Beschreibung
0..* : 0..1	Employee	Ein Projekt hat keinen oder einen Projektleiter.
0..* : 1	Employee	Ein Projekt hat genau einen definierten Projektmanager.
1 : 0..*	Worker	Ein Projekt hat beliebig viele Projektmitarbeiter.

Tabelle 20: Domainanalyse: Project Beziehungen

2.2.4 Worker

Dieses Konzept stellt die Mitarbeit eines Mitarbeiters innerhalb eines Projektes über einen bestimmten Zeitraum dar.

2.2.4.1 Attribute

Attribut	Beschreibung	Typ
startDate	Start der Mitarbeit	Date
endDate	Ende der Mitarbeit	Date

Tabelle 21: Domainanalyse: Worker Attribute

2.2.4.2 Beziehungen

Multiplizität	zu Konzept	Beschreibung
0..* : 1	Employee	Eine Mitarbeit in einem Projekt beinhaltet einen Mitarbeiter. Komposition auf Seite Mitarbeiter; Die Löschung des Mitarbeiters hat auch die Löschung der Mitarbeit im Projekt zur Folge.
0..* : 1	Project	Eine Mitarbeit in einem Projekt beinhaltet ein Projekt. Komposition auf Seite Projekt; Die Löschung des Projektes hat auch die Löschung der Mitarbeit im Projekt zur Folge.

Tabelle 22: Domainanalyse: Worker Beziehungen

2.2.5 Absence

Das Konzept Absence stellt eine Absenz eines Mitarbeiters dar.

2.2.5.1 Attribute

Attribut	Beschreibung	Typ
startTime	Beginn der Absenz	Date
endTime	Ende der Absenz	Date
state	Status der Absenz (Offen, Genehmigt, Abgelehnt)	enum
note	Beschreibung der Absenz	String
startDate	Beginn der periodischen Absenz	Date
endDate	Ende der periodischen Absenz	Date
period	Periodizität der Absenz (täglich, wöchentlich, monatlich, jährlich)	enum

Tabelle 23: Domainanalyse: Absence Attribute

2.2.5.2 Beziehungen

Multiplizität	zu Konzept	Beschreibung
0..* : 1	Employee	Eine Absenz gehört zu genau einem Mitarbeiter.
0..* : 1	AbsenceType	Eine Absenz ist immer von einem gewissen Absenztyp.

Tabelle 24: Domainanalyse: Absence Beziehungen

2.2.6 AbsenceType

Dieses Konzept beschreibt den Typ der Absenz (z.B. Ferien, Militär, oder Ähnliches).

2.2.6.1 Attribute

Attribut	Beschreibung	Typ
name	Bezeichnung des Typs.	String
color	Farbe des Typs.	String

Tabelle 25: Domainanalyse: AbsenceType Attribute

2.2.6.2 Beziehungen

Multiplizität	zu Konzept	Beschreibung
1 : 0..*	Absence	Ein Absenztyp kann von beliebig vielen Absenzen verwendet werden.

Tabelle 26: Domainanalyse: AbsenceType Beziehungen

3. Systemsequenzdiagramme

Anmerkung: Bei den CRUD Use Cases wird stellvertretend die Methode zur Erstellung (Create) erläutert.

3.1 UC1: Login

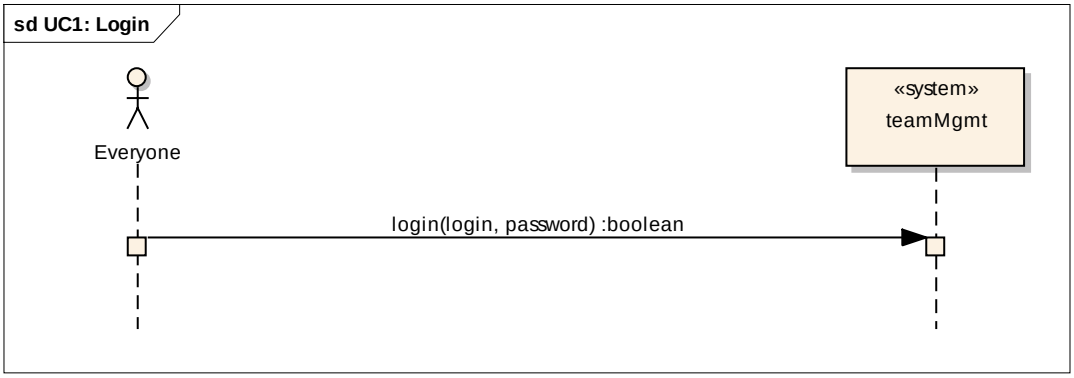

Abbildung 5: Domainanalyse: SSD UC1: Login

3.2 UC2: Change password

Abbildung 6: Domainanalyse: SSD UC2: Change Password

3.3 UC3: CRUD employee

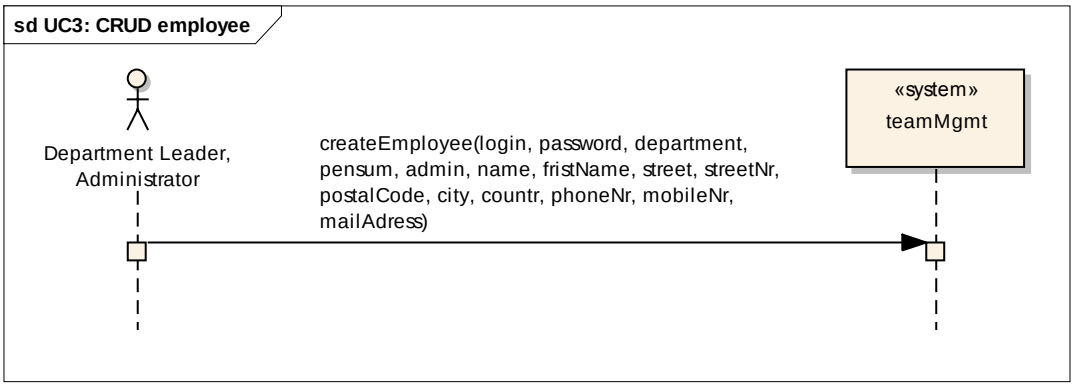

Abbildung 7: Domainanalyse: SSD UC3: CRUD Employee

3.4 UC4: CRUD absence

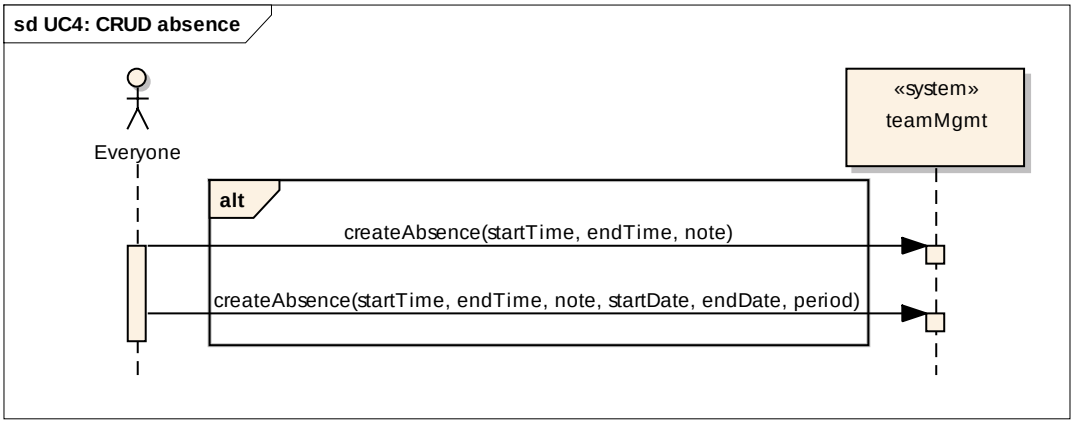

Abbildung 8: Domainanalyse: SSD UC4: CRUD absence

3.5 UC5: Approve/Disapprove absence

Abbildung 9: Domainanalyse: SSD UC5: Approve/Disapprove absence

3.6 UC6: CRUD department

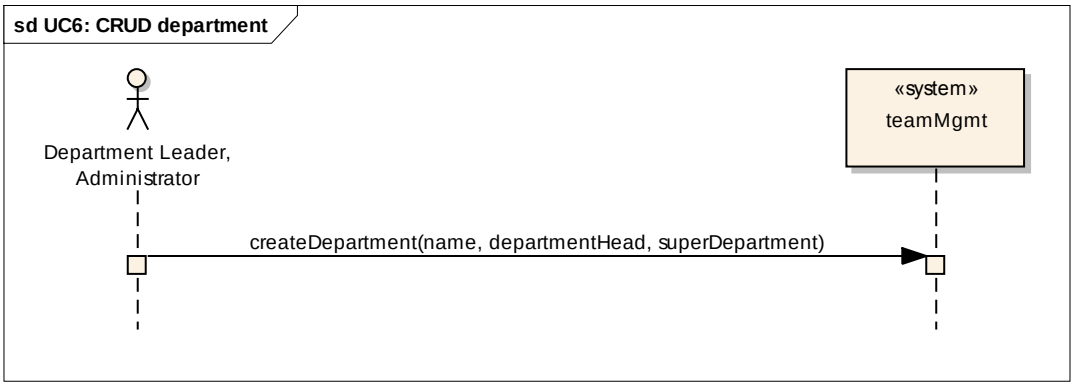

Abbildung 10: Domainanalyse: SSD UC6: CRUD department

3.7 UC7: CRUD project

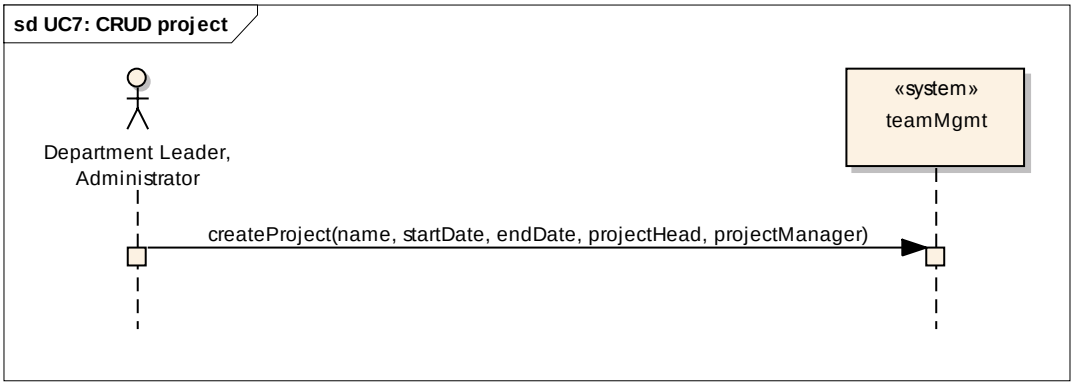

Abbildung 11: Domainanalyse: SSD UC7: CRUD project

3.8 UC8: Manage absence types

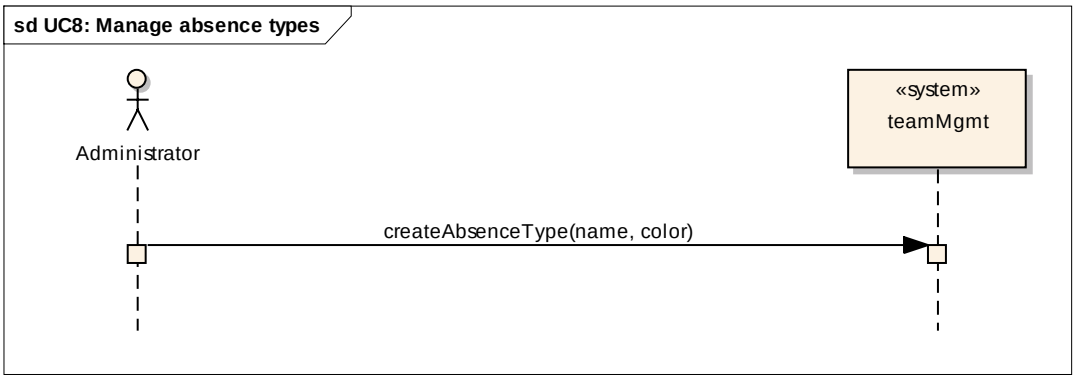

Abbildung 12: Domainanalyse: SSD UC8: Manage absence types

4. Systemoperationen

Anmerkung: Bei den CRUD Use Cases wird stellvertretend die die Methode zur Erstellung (Create) als Systemoperation erläutert.

4.1 Vertrag Systemoperation UC1: Login

Contract CO1: login	
Operation	login(login, password)
Querverweise	UC1: Login
Vorbedingung(en)	Mitarbeiter ist vorhanden.
Nachbedingung(en)	Eine neue Session wurde gestartet und Mitarbeiter ist eingeloggt im System.

Tabelle 27: Domainanalyse: Contract CO1: login

4.2 Vertrag Systemoperation UC2: Change password

Contract CO2: changePassword	
Operation	changePassword(oldPassword, newPassword1, newPassword2)
Querverweise	UC2: Change password
Vorbedingung(en)	- Mitarbeiter ist vorhanden. - Attribut oldPassword muss korrekt sein. - Attribute newPassword1 und newPassword2 müssen identisch sein.
Nachbedingung(en)	Das neue Passwort wurde im Mitarbeiter gespeichert.

Tabelle 28: Domainanalyse: Contract CO2: changePassword

4.3 Vertrag Systemoperation UC3: CRUD employee

Contract CO3: createEmployee	
Operation	createEmployee(login, password, department, pensum, admin, name, firstName, street, streetNr, postalCode, city, country, phoneNr, mobileNr, mailAdress)
Querverweise	UC3: CRUD employee
Vorbedingung(en)	Keine.
Nachbedingung(en)	- Ein neuer Mitarbeiter wurde erstellt. - Alle Attribute wurden korrekt initialisiert.

Tabelle 29: Domainanalyse: Contract CO3: createEmployee

4.4 Vertrag Systemoperation UC4: CRUD absence

Die zwei Arten von Absenzen werden auf unterschiedliche Arten erstellt. Diese werden in den folgenden Unterkapiteln erläutert.

4.4.1 Variante 1

Erstellung einer einmaligen Absenz.

Contract CO4-1:createAbsence	
Operation	createAbsence(startTime, endTime, note)
Querverweise	UC4: CRUD absence
Vorbedingung(en)	Mitarbeiter ist vorhanden.
Nachbedingung(en)	- Eine neue Absenz wurde erstellt und mit einem Mitarbeiter und einem Absenztyp verknüpft. - Das Attribut startTime ist kleiner als endTime. - Das Attribut state ist mit dem Wert für „offen“ initialisiert.

Tabelle 30: Domainanalyse: Contract CO4-1: createAbsence

4.4.2 Variante 2

Erstellung einer periodisch wiederkehrenden Absenz.

Contract CO4-2:createAbsence	
Operation	createAbsence(startTime, endTime, note, startDate, endDate, period)
Querverweise	UC4: CRUD absence
Vorbedingung(en)	Mitarbeiter ist vorhanden.
Nachbedingung(en)	- Eine neue Absenz wurde erstellt und mit einem Mitarbeiter und einem Absenztyp verknüpft. - Das Attribut startTime ist kleiner als endTime. - Das Attribut endDate grösser als endTime. - Das Attribut state ist mit dem Wert für „offen“ initialisiert.

Tabelle 31: Domainanalyse: Contract CO4-2: createAbsence

4.5 Vertrag Systemoperation UC5: Approve/Disapprove absence

Contract CO5: setAbsenceState	
Operation	setAbsenceState(state)
Querverweise	UC5: Approve/Disapprove absence
Vorbedingung(en)	Absenz ist vorhanden, mit dem Wert für „offen“ initialisiert und mit einem Mitarbeiter verknüpft.
Nachbedingung(en)	Das Attribut state ist mit einem zulässigen Wert für „genehmigt“ oder „abgelehnt“ initialisiert.

Tabelle 32: Domainanalyse: Contract CO5: setAbsenceState

4.6 Vertrag Systemoperation UC6: CRUD department

Contract CO6: createDepartment	
Operation	createDepartment(name, departmentHead, superDepartment)
Querverweise	UC6: CRUD department
Vorbedingung(en)	Keine.
Nachbedingung(en)	Eine neue Abteilung wurde erstellt und falls nötig korrekt mit dem Abteilungsleiter und der Superabteilung verknüpft.

Tabelle 33: Domainanalyse: Contract CO6: createDepartment

4.7 Vertrag Systemoperation UC7: CRUD project

Contract CO7: createProject	
Operation	createProject(name, startDate, endDate, projectHead, projectManager)
Querverweise	UC7: CRUD project
Vorbedingung(en)	Keine.
Nachbedingung(en)	- Ein neues Projekt wurde erstellt, mit dem Projektmanager und falls nötig korrekt mit dem Projektleiter korrekt verknüpft. - Das Attribut startDate ist kleiner als endDate.

Tabelle 34: Domainanalyse: Contract CO7: createProject

4.8 Vertrag Systemoperation UC8: Manage absence types

Contract CO8: manageAbsenceTypes	
Operation	manageAbsenceTypes(name, color)
Querverweise	UC8: Manage absence type
Vorbedingung(en)	Keine.
Nachbedingung(en)	Ein neuer Absenztyp wurde erstellt.

Tabelle 35: Domainanalyse: Contract CO8: manageAbsenceTypes

Technischer Bericht

1. Evaluation Java EE Technologien

1.1 Applikationsserver

Ein Applikationsserver ist im Allgemeinen ein Server, auf welchem Anwendungsprogramme ausgeführt werden. Das Produkt teamMgmt verwendet einen solchen Applikationsserver als Grundlage. Folgende Applikationsserver wurden als mögliche Kandidaten eingestuft: JBoss, GlassFish, WebLogic und Websphere.

Die Produkte WebLogic und Websphere sind allerdings kostenpflichtig und zu teuer für dieses Projekt mit den gegebenen verfügbaren Mitteln. Aus diesem Grund werden in den folgenden Abschnitten nur die zwei Server JBoss und GlassFish detaillierter beschrieben und nach gewissen Kriterien bewertet. Es handelt sich um eine Evaluation, welche zu Beginn des Projektes durchgeführt wurde, und schliesslich zur Auswahl eines geeigneten Applikationsservers führte.

1.1.1 Typen

1.1.1.1 JBoss

1.1.1.1.1 Allgemein

Das Open Source Projekt JBoss existiert seit 1999. Die Abkürzung JBoss steht für Java Beans Open Source Software. Die heutige JBoss Incorporation, welche zu Red Hat gehört, kümmert sich um die Weiterentwicklung des JBoss Enterprise Middelware Suite.

Diese beinhaltet unter Anderen folgende Teilprojekte:

- JBoss Application Server
- JBoss Eclipse Tools
- Hibernate (JBoss JPA Integration)
- usw.

Die JBoss Enterprise Middelware Suite ist grösstenteils Open Source und frei erhältlich. Red Hat bietet einen breites Supportangebot, welches jedoch kostenpflichtig ist.

JBoss ist momentan einer der meistverbreitetesten Applikationsserver, es existieren zwar keine genauen Zahlen, jedoch kann aufgrund der Webpräsenz, sowie der Download Zahlen darauf geschlossen werden

1.1.1.1.2 Funktionalität

Die folgende Aufzählung beinhaltet nur einen Auszug der Funktionalitäten. Es wurde hauptsächlich die für das Projekt teamMgmt relevanten Funktionen und unterstützten Technologien betrachtet. Dabei handelt es sich um die aktuelle JBoss Version 7.1.

Funktionalität	Bemerkung
Plattform	Ausser Mac OS werden alle gängigen Plattformen (Windows, Linux und Solaris) unterstützt
Java EE Version	Unterstützt Java EE Version 6
Java Web Technologien	Unterstützt Servlet 3.0, JSP 2.2 und JSF 2.0
Next Generation Web	Unterstützt Ajax Framework
Enterprise Application Technologien	Unterstützt EJB 3.1 und JPA 2.0
Datenbanksysteme	Unterstützt DB2, Oracle, PostgreSQL, Microsoft SQL Server und MySQL Server
Entwicklungsumgebung	Eclipse und JBoss Developer Studio

Tabelle 36: Technischer Bericht: JBoss Funktionalitäten

1.1.1.1.3 Administration

Inbetriebnahme

Um die Installation von JBoss auszuführen, gibt es nur eine Variante. Man muss sich das Datenarchiv von der JBoss Community herunterladen und in den gewählten Installationszielort entpacken. Anschliessend müssen noch eine Umgebungsvariable und der Systempfad angepasst werden und der Applikationsserver kann gestartet werden.

Management

Die Management Konsole, wirkt etwas spärlich. Sie ist in die zwei Bereiche Profile und Runtime unterteilt. Runtime bietet diverse Auswertungsmöglichkeiten der verfügbaren Ressourcen. Applikationen können dort hinzugefügt und entfernt werden. Der Bereich Profile bietet die Konfigurationsmöglichkeiten des Servers in den Bereichen Core, Connector, Container, Security und Web. Es können auch grundlegende Einstellungen wie das Binding der Sockets oder Interfaces vorgenommen werden.

1.1.1.1.4 Fazit

Die Funktionalitäten decken alle Anforderungen, welche das Projekt teamMgmt hat, ab. Die Integration in Eclipse ist einfach möglich und gut anwendbar. JBoss ist ein sehr umfangreiches Tool, dessen Möglichkeiten kaum vom Projekt teamMgmt ausgenutzt würden.

1.1.1.2 GlassFish

1.1.1.2.1 Allgemein

Der Applikationsserver GlassFish gehört dem GlassFish Project an, welches von Oracle betrieben wird. Mit dem Beginn des Open Source Projektes bildete sich eine Gemeinschaft, welche zusammen mit Oracle das GlassFish Projekt betreuen und pflegen.

Bei GlassFish handelt es sich um die Referenzimplementierung der Java EE Plattform.

1.1.1.2.2 Funktionalität

Die folgende Aufzählung beinhaltet nur einen Auszug der Funktionalitäten. Es wurde hauptsächlich die für das Projekt teamMgmt relevanten Funktionen und unterstützten Technologien betrachtet. Dabei handelt es sich um die aktuelle GlassFish Version 3.1.

Funktionalität	Bemerkung
Plattform	Alle gängigen Plattformen (Windows, Unix, Solaris und Mac OS) werden unterstütz
Java EE Version	Unterstützt Java EE Version 6
Java Web Technologien	Unterstützt Servlet 3.0, JSP 2.2 und JSF 2.0
Next Generation Web	Unterstützt Ajax Framework
Enterprise Application Technologien	Verwendet eine Java DB (10.5.3.0) und unterstützt EJB 3.1 und JPA 2.0
Datenbanksysteme	Unterstützt DB2, Oracle, Microsoft SQL Server und MySQL Server
Entwicklungsumgebung	Eclipse und NetBeans

Tabelle 37: Technischer Bericht: GlassFish Funktionalitäten

1.1.1.2.3 Administration

Inbetriebnahme

Um mit der Installation von GlassFish zu beginnen, muss mindestens Java EE Version 6 installiert sein. Es kann ein komprimiertes Datenarchiv oder ein Installer zur Installation unter Windows verwendet werden. Ausserdem müssen gegebenenfalls eine Umgebungsvariable und der Systempfad angepasst werden. Mittels dem GUI Installer wird man schrittweise durch die Installation geführt und kann bereits einige Konfigurationseinstellungen vornehmen. Anschliessend kann der Applikationsserver gestartet werden.

Management

Die Management Konsole von GlassFish ist übersichtlich aufgebaut. Sie bietet diverse Möglichkeiten zur Auswertung der verfügbaren Ressourcen. Es können neue Serverinstanzen erstellt und den Applikationen hinzugefügt werden.

1.1.1.2.4 Fazit

GlassFish ist ein wenig schlanker als JBoss, enthält jedoch auch alle für das Projekt teamMgmt benötigten Funktionalitäten. Dies und die übersichtlichere Management Konsole sind klare Pluspunkte für GlassFish. Die Integration in Eclipse ist ebenfalls einfach möglich und gut anwendbar.

1.1.2 Gegenüberstellung

1.1.2.1 Entscheidungskriterien

In den folgenden Abschnitten werden die Entscheidungskriterien kurz beschrieben, welche dann im Kapitel *Übersicht* zur Bewertung der einzelnen Technologien verwendet werden.

1.1.2.1.1 Unterstützte Technologien

Die aktuelle Version von Java EE (Version 6) und zusätzliche Web Technologien, wie JSF 2.0 und Servlet 3.0 sollten unterstützt werden.

1.1.2.1.2 Integration Entwicklungsumgebung

Eine Integration in Eclipse, welches vom Projekt teamMgmt verwendet wird, sollte leicht möglich und brauchbar in der Anwendung sein. Eine vom Hersteller mitgelieferte Entwicklungsumgebung könnte auch von Nutzen für das Projekt sein, da sie auf den Kontext bezogen entwickelt wurde.

1.1.2.1.3 Persistenz

Hierbei wird die Art der Integration der Datenbank genauer bewertet. Ist eine direkte Integration über JPA möglich oder muss zum Beispiel mit Hibernate gearbeitet werden.

1.1.2.1.4 Unterstützte Datenbanksysteme

Das Projekt teamMgmt verwendet eine Datenbank. Welche sind integriert, respektive werden vom Applikationsserver unterstützt werden.

1.1.2.1.5 Inbetriebnahme & Management

Die Inbetriebnahme und das Aufsetzen des Applikationsservers sollten einfach und rasch durchführbar sein. Ausserdem sollte überprüft werden ob ein Neustart oder allgemein das Management des Servers ohne grosse Umstände möglich ist.

1.1.2.1.6 Verbreitung & Zukunftsaussichten

Der Applikationsserver sollte weitverbreitet sein, dass auch in absehbarer Zukunft sichergestellt ist, dass das Produkt nicht vom Markt verschwindet. Zudem sollte die Technologie auch in Zukunft noch gut unterstützt und weiterentwickelt werden, so dass auch das Produkt teamMgmt die Möglichkeit hat einfacher weiterentwickelt zu werden.

1.1.2.2 Übersicht

		JBoss			GlassFish		
Kriterium	**Gew.**	**Wert.**	**Beschreibung**	**Tot.**	**Wert.**	**Beschreibung**	**Tot.**
1.1.2.1.1 Unterstützte Technologien	10	10	Alle geforderten Technologien werden unterstützt	100	10	Alle geforderten Technologien werden unterstützt	100
1.1.2.1.2 Integration Entwicklungsumgebung	6	10	Einfach möglich und gut anwendbar	60	9	Einfach möglich und gut anwendbar	54
1.1.2.1.3 Persistenz	5	10	JPA ist möglich	50	10	JPA ist möglich	50
1.1.2.1.4 Unterstützte Datenbanksysteme	8	10	HSQL integriert, MySQL unterstützt	80	10	Java DB integriert MySQL wird unterstützt	80
1.1.2.1.5 Inbetriebnahme und Management	7	7	Einfache Inbetriebnahme und jedoch spärliches Management	49	10	Einfache Inbetriebnahme und übersichtliches Management	70
1.1.2.1.6 Verbreitung & Zukunftsaussichten	5	9	Meistverbreitester AS, ständige Entwicklung erkennbar	45	9	Aufstrebend, weitverbreitet, ständige Entwicklung erkennbar, Referenzimpl. Java EE	45
TOTAL				**384**			**399**

Tabelle 38: Technischer Bericht: Bewertungstabelle

1.1.2.3 Auswahl

Beide Technologien wären für das Projekt teamMgmt sehr gut anwendbar. JBoss ist jedoch in Sachen Management etwas umständlicher als GlassFish. Ausserdem ist die Technologie von JBoss sehr umfangreich, was in Bezug auf das Produkt teamMgmt nicht nötig ist und die ganze Sache nur komplizierter macht. Auf Grund der knappen Entscheidung nach den gewählten Kriterien fällt die Auswahl auf **GlassFish**. Falls bei der Implementierung jedoch andere unerwartete Probleme auftauchen, wäre ein Wechsel zu JBoss ohne grosse Überlegungen möglich. Dies weil beide Technologien ähnlich gut sind und unter Umständen erst die konkrete Anwendung allfällige Mängel aufzeigen könnte.

1.2 User Interface Komponente

Die Wahl eines Komponentenframeworks zur Unterstützung von JSF bei der Erstellung von User Interfaces kam auf Primefaces. Ein weiteres Framework, welches zur eventuellen Auswahl stand, ist Richfaces. Dieses bietet ebenfalls anschauliche Komponenten und ist ein Projekt von JBoss. Da Primefaces innerhalb der Vorlesung von Internettechnologien vorkommt und das Projektteam mit diesem Framework schon etwas Erfahrung hatte, fiel die Auswahl auf Primefaces.

Die Auswahl einer User Interface Komponente war nicht von hoher Wichtigkeit, deshalb wurde mehr Wert auf die Auswahl eines geeigneten Applikationsservers gelegt.

2. Technologische Aspekte

Der Java EE Framework Standard JSF wird in der aktuellsten Version 2 verwendet. Dies in Kombination mit der neusten Version von Primefaces könnte zu möglichen Kompatibilitätsproblemen führen. Da beide Elemente zur Erstellung von HTML Seiten anbieten, kann es sein das einzelne Komponenten der einen Technologie mit solchen der Anderen nicht kompatibel ist. Ausserdem ist es möglich, dass gewisse Verschachtelungen von Komponenten der verschiedenen Technologien Probleme aufweisen.

Die aktuellste Version des GlassFish Servers unterstützt JSF 2.0 vollumfänglich. Eine mögliche Schwierigkeit, welche hier auftrat, war wiederum das Zusammenspiel mit Primefaces. Gewisse Elemente von Primefaces werden in Konfigurationsdateien definiert, was eventuell vom GlassFish Server nicht verstanden werden könnte.

Probleme, welche im Zusammenhang mit den Technologien auftauchten, sind in den folgenden Abschnitten beschrieben

2.1 JSF Template & Primefaces

2.1.1 Problem

Um JSF Seiten für eine Webapplikation logisch aufzubauen werden Komponenten zur Erstellung von Templates angeboten. Man muss dazu eine XHTML Seite erstellen, welche als Template dienen soll. Sie definiert den Grundaufbau aller Seiten und ermöglicht es so den Entwicklern jeweils nur den Inhalt neu laden zu müssen (ohne Kopfzeile und Navigation).

Ein Beispiel für den Aufbau von Templates mit Hilfe von JSF wird in den folgenden zwei Abbildungen aufgezeigt.

```
<h:head>
    <title><ui:insert name="title"/></title>
</h:head>

<h:body>
    <table>
        <tr>
            <td colspan="2">
                <ui:include src="header.xhtml"/>
            </td>
        </tr>
        <tr>
            <td>
                <h:panelGroup id="navigation">
                    <ui:include src="navigation.xhtml"/>
                </h:panelGroup>
            </td>
            <td>
                <h:panelGroup id="content">
                    <ui:insert name="content"/>
                </h:panelGroup>
            </td>
        </tr>
    </table>
</h:body>
```

Abbildung 13: Technischer Bericht: JSF Template (1/2)

```
<ui:composition template="template.xhtml">
    <ui:define name="title">title of application</ui:define>
    <ui:define name="content">
        ...
    </ui:define>
</ui:composition>
```

Abbildung 14: Technischer Bericht: JSF Template (2/2)

Die zweite Abbildung zeigt wie ein Template anschliessend vervollständigt wird, in dem man Elemente definiert und an Stelle der „…“ den Inhalt der Seite programmiert.

Das Problem das nun auftauchte, war, dass Komponenten von Primefaces innerhalb einer ui:composition Komponente weder erkannt noch dargestellt wurde. Bei einer ausführlichen Suche nach diesem Mangel im Internet konnte keine Lösung gefunden werden, welche den Template Mechanismus von JSF verwendet.

2.1.2 Lösung

Es konnten diverse Lösungen mit Alternativen für den Template Mechanismus von JSF gefunden werden. Folgende Alternative wurde implementiert und konnte als ebenbürtig erachtet werden.

```
<h:head>
    <title>teamMgmt</title>
</h:head>

<h:body>
    <table>
        <tr>
            <td colspan="2">
                <h:graphicImage value="res/images/header.png"/>
            </td>
        </tr>
        <tr>
            <td>
                <h:panelGroup id="navigation">
                    <ui:include src="navigation.xhtml"/>
                </h:panelGroup>
            </td>
            <td>
                <h:panelGroup id="content">
                    <ui:include src="#{navigationControl.pageName}"/>
                </h:panelGroup>
            </td>
        </tr>
    </table>
</h:body>
```

Abbildung 15: Technischer Bericht: JSF Template Alternative (1/3)

Bei „navigationControl“ handelt es sich um ein Managed Bean für eine aktive Session. Den Aufbau dieses Beans wird in folgender Abbildung aufgezeigt.

```
public class NavigationController {
    private String pageName = "login.xhtml";

    public String getPageName() {
        return pageName;
    }

    public void login() {
        pageName = "login.xhtml";
    }

    public void start() {
        pageName = "start.xhtml";
    }

    public void newEmployee() {
        pageName = "employee/employeeNew.xhtml";
    }
```

Abbildung 16: Technischer Bericht: JSF Template Alternative (2/3)

Die einzelnen Aufrufe der Methoden innerhalb des NavigationControllers sind in folgender Abbildung ersichtlich.

```
<h:form>
    <p:menu>
        <p:submenu label="Mitarbeiter">
            <p:menuitem value="Neu" action="#{navigationControl.newEmployee}" update=":content"/>
            <p:menuitem value="Meine Daten" action="#{navigationControl.editEmployee}" update=":content"/>
            <p:menuitem value="Suchen" action="#{navigationControl.searchEmployee}" update=":content"/>
        </p:submenu>
    </p:menu>
</h:form>
```

Abbildung 17: Technischer Bericht: JSF Template Alternative (3/3)

Durch den Aufruf einer Methode innerhalb des NavigationControllers wird ein neuer Seitenname gesetzt und das PanelGroup „content“ wird mit dem neuen Inhalt aktualisiert und neu geladen. Es tauchte jedoch ein weiteres Problem auf, welches in folgendem Abschnitt beschrieben wird.

Mit dieser Variante konnte eine passable Lösung implementiert werden, welche nun die Erstellung von Komponenten von Primefaces ermöglicht und es trotzdem erlaubt pro Seite nur den Inhalt definieren zu müssen und Kopfzeile / Navigation nur einmal zu laden.

2.2 Doppelklick

2.2.1 Problem

Nach der Lösung des Template Problems tauchte das Problem des Doppelklicks innerhalb der Menüs auf. Man benötigte um den Aufruf einer neuen Seite zu erhalten zwei Klicks auf den jeweiligen Menüpunkt. Dies rührt daher, das die Phasen von JSF anders ablaufen als für diese Lösungsvariante notwendig. Ein Menüeintrag wurde folgendermassen definiert:

```
<p:menuitem value="Neu" action="#{navigationControl.newEmployee}" update=":content"/>
```

Mittels dem update wird das PanelGroup des Inhalts neu geladen und mittels Action wird der Seitenname neu gesetzt, der innerhalb dieses PanelGroups eingefügt wird. Die Lebenszyklus Phasen von JSF sind in der folgenden Abbildung dargestellt.

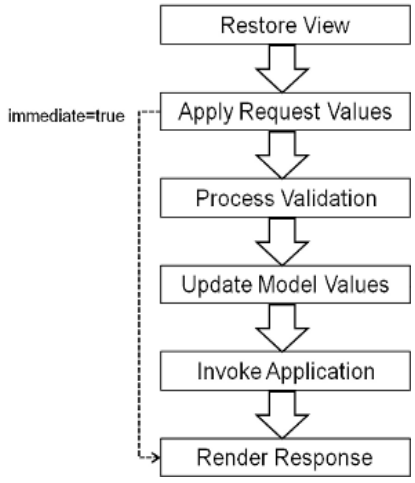

Abbildung 18: Technischer Bericht: JSF Lebenszyklus

Der update Befehl wird bereits in Phase 1 (Restore View) ausgeführt und der action Befehl kann frühestens in Phase 2 (Apply Request Values) oder normalerweis in Phase 5 (Invoke Application) ausgeführt werden. Somit wird der Seitennamen zwar neu gesetzt, aber die Seite wird zu früh neu geladen. Dies hat zur Folge, dass Änderungen erst beim zweiten Ausführen derselben Aktion sichtbar werden.

2.2.2 Lösung

Um eine Aktion innerhalb des Menüs bereits beim ersten Klick auszuführen, muss innerhalb der Aktion ein Forward ausgeführt werden damit die Applikation eine Antwort generiert und Phase 6 (Render Response) diese an den Benutzer zurückgibt. Zu diesem Zweck wurde ein globaler Forward implementiert. Den Einbau in den NavigationController wird in folgender Abbildung ersichtlich.

```
public class NavigationController {
    private String pageName = "login.xhtml";
    private String globalForward = "success";

    public String getPageName() {
        return pageName;
    }

    public String login() {
        pageName = "login.xhtml";
        return globalForward;
    }

    public String start() {
        pageName = "start.xhtml";
        return globalForward;
    }

    public String newEmployee() {
        pageName = "employee/employeeNew.xhtml";
        return globalForward;
    }
```

Abbildung 19: Technischer Bericht: Doppelklick Alternative (1/2)

Dieser Forward muss noch in der Faces Konfiguration definiert werden.

```
<navigation-rule>
    <from-view-id>/content/master.xhtml</from-view-id>
    <navigation-case>
        <from-outcome>success</from-outcome>
        <to-view-id>/content/master.xhtml</to-view-id>
    </navigation-case>
</navigation-rule>
```

Abbildung 20: Technischer Bericht: Doppelklick Alternative (2/2)

Somit wird immer auf dieselbe Seite weitergeleitet, jedoch werden jeweils nur die Elemente neu geladen, welche sich verändert haben, was in diesem Falle dem Inhalt der neuen Seite entspricht.

2.3 Schedule Komponente

2.3.1 Problem

Für die Anzeige aller Absenzen in der Übersicht wird die Schedule Komponenten von Primefaces verwendet. Das Problem hierbei ist, dass diese Komponente, solange sie sich innerhalb eines Panels (ebenfalls von Primefaces) befindet, beim erstmaligen Aufruf nicht direkt dargestellt wird. Um den Kalender anzeigen zu können muss ein Neu laden der Seite (F5) veranlasst werden.

2.3.2 Lösung

Bei der Auswahl zur Ansicht der Übersicht wird diese nicht direkt geladen sondern die Auswahlliste für die Abteilungen wird angezeigt, wie in folgender Abbildung dargestellt.

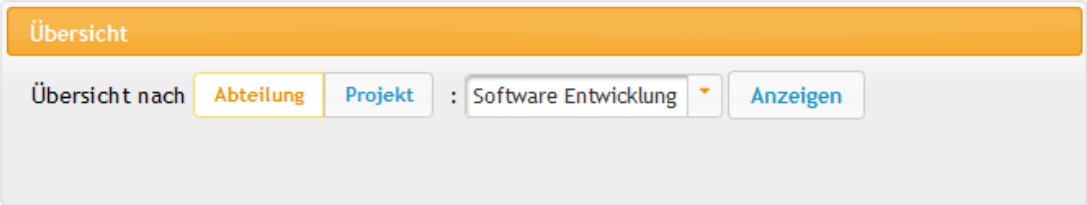

Abbildung 21: Technischer Bericht: Schedule Komponente Alternative (1/2)

Der Benutzer hat nun die Möglichkeit zuerst die gewünschte Abteilung oder das gewünschte Projekt auszuwählen und anschliessend durch wählen des *Anzeigen* Buttons wird die Schedule Komponente geladen. Die Funktion des *Anzeigen* Buttons wird in folgender Abbildung ersichtlich.

```
...

<p:commandButton value="Anzeigen" action="..."
                 oncomplete="javascript:location.reload(true)" update="schedule"/>

...

<h:panelGroup id="schedule">
    ...
</h:panelGroup>
```

Abbildung 22: Technischer Bericht: Schedule Komponente Alternative (2/2)

Das Wählen des Anzeigen Buttons hat die Ausführung einer Aktion zur Folge, welche alle Absenzen aller Mitarbeiter der gewünschten Abteilung oder des gewünschten Projektes lädt. Ausserdem wird eine panelGroup, welche die effektive Schedule Komponente beinhaltet, neu geladen. Bei Fertigstellung dieser Aktionen wird die gesamte Lokation neu geladen, verhält sich gleich wie das neue laden einer Seite im Webbrowser. Dies hat zur Folge, dass die Schedule Komponente zur Ansicht kommt.

Bei weiteren Versuchen wäre dieses neu laden der gesamten Lokation nicht notwendig, beim ersten Laden ist es jedoch unumgänglich. Die Applikation erhält dadurch weder quantitative noch qualitative Einbussen.

2.4 Entity Manager (Factory)

2.4.1 Problem

Zu Beginn der Arbeit wurde der zentrale PersistenceService erstellt, welcher alle Zugriffe auf die Datenbank steuerte. Zuerst wurde ein einzelner Entity Manager verwendet, welcher bei der Instanziierung erstellt und danach verwendet wird. Da aber mehrere Threads gleichzeitig auf den Service zugreifen entstanden Fehler und die Applikation stürzte ab. Ausserdem konnte bei einem Synchronisieren des Servers mit der Applikation anschliessend die Software nicht mehr ausgeführt werden, da immer noch ein bestehender EntityManager geöffnet war. Dies erforderte ein ständiges Stoppen und Starten des Servers während der Entwicklung.

2.4.2 Lösung

Daher ist es besser in jeder Methode nach Bedarf einen EntityManager zu erstellen und in einem finally-Block wieder zu schliessen. Dieses Verfahren wird an Hand der getEntityById Methode in der folgenden Abbildung sichtbar.

```
public <T> T getEntityById(Integer id, Class<T> c) throws PersistenceException {
    EntityManager entityManager = null;
    try {
        entityManager = createEntityManager();
        return entityManager.find(c, id);
    } catch (PersistenceException pe) {
        throw pe;
    } catch(Exception e) {
        throw new PersistenceException(e);
    } finally {
        closeEntityManager(entityManager);
    }
}
```

Abbildung 23: Technischer Bericht: EntityManager

3. Fachliche Aspekte

3.1 Rechte & Rollen

3.1.1 Problemstellung

Die Verteilung der Rollen und der dazugehörenden Rechte konnte ebenfalls schon früh als mögliches Problem festgestellt werden. Da die Anforderungen vom Team eingebracht werden musste man sich hierzu einige Gedanken machen. Folgende Fragen mussten vorab geklärt werden um während der Implementierung nicht zusätzliche Probleme zu schaffen:

- Wer kann neue Mitarbeiter erstellen und kann er diese in jede beliebige Abteilung setzen?
- Wer kann neue Abteilungen erstellen und wo darf er diese angliedern?
- Wer kann neue Projekte erstellen?
- Wer kann welche Mitarbeiter zu Projekten hinzufügen?
- Wer kann Projektleiter bestimmen und ändern?
- Wer kann Typen von Absenzen festlegen?
- Kann ein Projektleiter Absenzen genehmigen oder nur Abteilungsleiter?
- Wer kann in einer Übersicht Absenzen sehen und inwiefern sind diese eingeschränkt?

Diese und noch viele weitere Fragen wurden zum Teil vor Beginn der Implementierung geklärt oder tauchten während der Implementierung auf.

3.1.2 Lösung

Grundsätzlich wird für jeden Mitarbeiter (Benutzer) sobald er sich einloggt eine Objekt der Klasse Role erstellt, welches anschliessend aufgesetzt wird und während der aktiven Session gespeichert wird. Dieses Objekt wird bei Bedarf neu aufgesetzt, sofern Änderungen innerhalb der Strukturen von Abteilungen stattfinden oder ähnliche Änderungen, welche die Rolle beeinflussen. Die folgende Abbildung zeigt den Aufbau der Klasse.

```
public class Role {
    private boolean admin;

    private ArrayList<Integer> departmentIds;
    private ArrayList<Integer> projectIds;
    private ArrayList<Integer> allProjectIds;

    ...
}
```

Abbildung 24: Technischer Bericht: Rollen & Rechte Alternative (1/2)

Die Rolle interessiert sich nur, ob ein Mitarbeiter ein Administrator ist oder nicht. Aufgrund der einzelnen Listen innerhalb der Klasse werden dann die Identifikationen der erlaubten Abteilungen oder Projekte geladen. Dieses Setup der Rolle wird in folgender Codesequenz aufgezeigt.

```
public void setUpRole(String login) throws HandlerException {
    employee = EmployeeHandler.getInstance().getEmployeeByLogin(login);
    role = new Role();

    if (employee.isAdmin()) {
        role.setAdmin(true);
        addAllDepartmentIds();
        addAllProjectIds();
    } else {
        addDepartmentIdsForHeadToRole();
    }

    addProjectIdsForHeadOrManagerToRole();
}
```

Abbildung 25: Technischer Bericht: Rollen & Rechte Alternative (2/2)

Zuerst wird der entsprechende Mitarbeiter mittels seines Benutzernamen (login) von der Datenbank geladen und ein neues Objekt der Klasse Role wird angelegt. Handelt es sich um einen Administrator so werden die Listen der erlaubten Abteilungen und Projekte mit allen verfügbaren Identifikationen gefüllt, da ein Administrator alles verwalten kann. Falls es sich um keinen Administrator handelt werden die einzelnen erlaubten Abteilungen und Projekte an Hand der Identifikation des Mitarbeiters in die entsprechenden Listen geladen.

Diese Listen dienen anschliessend innerhalb der User Interfaces oder der Managed Beans zur Überprüfung der Rolle des aktuell eingeloggten Mitarbeiters. Das Objekt der Klasse Role wird in einem Session Bean abgespeichert, welches innerhalb jedes User Interfaces zur Verfügung steht. Über Methoden mit booleschen Rückgabewerten werden die einzelnen Komponenten des User Interface deaktiviert oder aktiviert und die entsprechend erlaubten Auswahllisten können geladen werden.

3.2 Konfigurierbarkeit

3.2.1 Problemstellung

Bei solchen Applikationen mit komplexen Rollen und Rechten taucht die Frage auf, inwiefern die Applikation konfigurierbar sein soll. Die Anforderungen und Möglichkeiten innerhalb der Applikation sind vom Projektteam definiert worden. Jegliche Konfigurationsmöglichkeit dieser Einstellungen macht die Implementierung komplexer und wurde deshalb vorerst aus dem Prozess ausgeschlossen. Es wurde versucht darauf zu achten eine zukünftige Konfigurierbarkeit möglich zu machen. Diese Variante wird jedoch nur ausgeführt, sofern noch genügend Zeit am Ende des Projektes zur Verfügung steht. Ansonsten bleibt dies eine Idee für die Weiterentwicklung der Software.

3.2.2 Effektive Konfigurierbarkeit

Es wurde schlussendlich, auch aus Zeitgründen, nur die Verwaltung von Absenztypen und deren Farbe zur Anzeige in der Übersicht konfigurierbar gemacht. Dies kann jedoch nur von einem Administrator durchgeführt werden. Die folgende Abbildung zeigt die Verwaltung der Absenztypen.

Abbildung 26: Technischer Bericht: Konfigurierbare Absenztypen

Software Architektur

1. Einführung

1.1 Zweck

Dieses Dokument beschreibt die Architektur der Software teamMgmt.

1.2 Gültigkeitsbereich

Die Gültigkeit des vorliegenden Software Architektur Dokumentes erstreckt sich über das gesamte Projekt teamMgmt bis hin zum geplanten Projektende des 15. Juni 2012.

1.3 Referenzen

Nachfolgend werden die Dokumente aufgeführt, welche von diesem Dokument referenziert werden:

- Projektplan.pdf
- Anforderungsspezifikation.pdf
- Domainanalyse.pdf
- Technologiestudie.pdf
- Glossar.pdf

1.4 Übersicht

Im folgenden Kapitel *Systemübersicht* werden die Softwarearchitektur des Systems und deren einzelne Elemente aufgezeigt und beschrieben.

Softwareanforderungen und Objekte, welche einen Einfluss auf die Architektur haben werden im Kapitel *Architektonische Ziele & Einschränkungen* umschrieben. Ausserdem werden das Design und die Implementationsstrategie definiert.

Die Festlegung der logischen Struktur des Produktes folgt im Kapitel *Logische Architektur*.

Die einzelnen Prozesse und die Komponenten und deren Aufteilung werden im Kapitel *Prozesse und Threads* und *Deployment* angegeben.

In den letzten Kapiteln *Datenspeicherung* und *Grössen und Leistung* wird das Datenmodell der Datenbank aufgezeigt und die Einschränkungen der Applikation bezüglich Speicher und Leistung werden beschrieben.

2. Systemübersicht

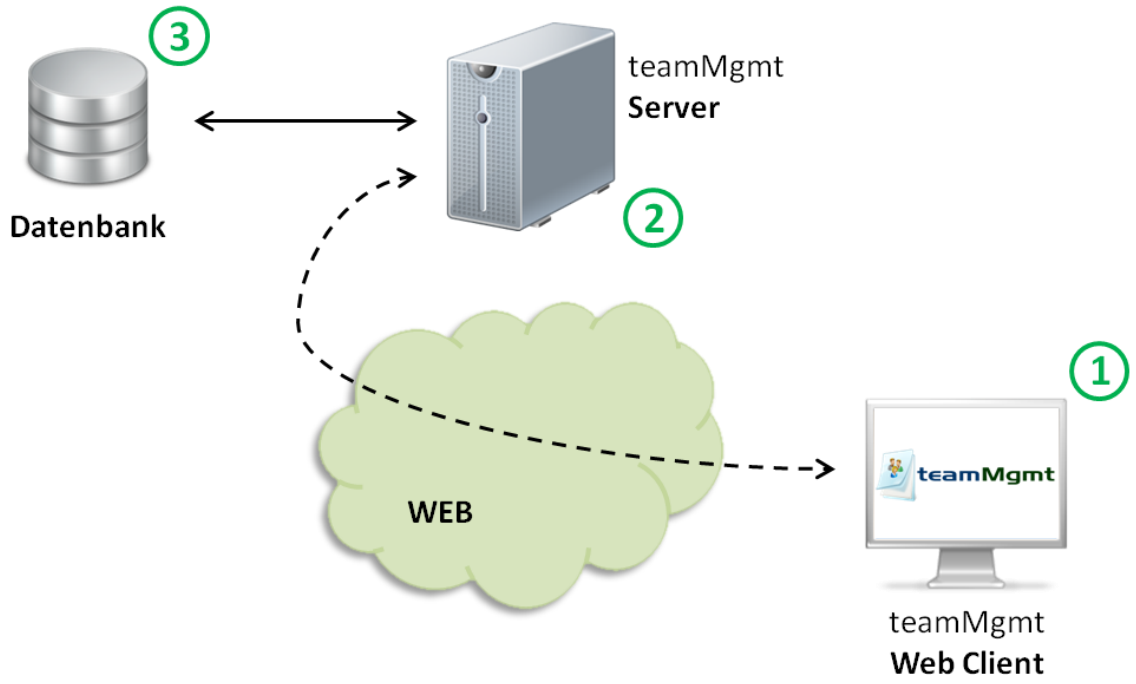

Abbildung 27: Software Architektur Dokument: Systemarchitektur

Die Java EE Applikationsarchitektur ist typischerweise in vier Tiers unterteilt. Für das Produkt teamMgmt verwenden wir jedoch eine 3-Tier-Architektur. Als erster Tier ist der Client Tier definiert, welcher auf dem Rechner des Benutzers angesiedelt ist und in diesem Falle dem Webbrowser entspricht. Die ursprünglichen Tiers 2 und 3, Business und Web Tiers sind Bestandteil des Java EE Servers, im Falle vom Produkt teamMgmt ist dies ein GlassFish Applikationsserver, welcher diese beiden Tiers verbindet. Diese könnten auch getrennt auf verschiedenen Servern laufen, was jedoch nur mit Änderungen innerhalb des Source Codes möglich wäre. Der letzte Tier ist der EIS Tier, welcher den Datenbankserver beinhaltet. Die einzelnen Tiers und deren Umsetzung werden in den folgenden Abschnitten genauer beschrieben.

2.1.1 Client Tier

Die vom JSF Framework generierten Views werden via http an den Client übermittelt und dort im Webbrowser dargestellt. Die Eingaben des Nutzers werden als Post Nachrichten an den Server übermittelt und dort verarbeitet. Die Clients benötigen kein eigenes Deployment Package. Aufrufe an die Applikation erfolgen über einen beliebigen Webbrowser, welche vom Applikationsserver behandelt werden.

2.1.2 Web Tier

Dieser Tier wird auf dem GlassFish Server ausgeführt. Er beinhaltet die Java Server Faces (JSF) und deren Managed Beans. In den JSF Managed Beans ist der Grossteil der Geschäftslogik untergebracht. Sie werden zum Einen zur Steuerung der User Interfaces, wie auch zur Weitergabe der Interaktionen eines Benutzers an tiefere Schichten verwendet.

2.1.3 Business Tier

Der Business Tier ist ebenfalls auf dem GlassFish Server angesiedelt. Bei teamMgmt werden dort der Datenbankzugriff, die Persistierung und Teile der Geschäftslogik behandelt. Einerseits werden EJBs, als Entities, zur Abbildung der Domainobjekt verwendet, welche mit Hilfe von JPA (Java Persistence API) in der Datenbank gespeichert werden. Die Daten der EJBs werden in Domainobjekte geladen, welch danach im Web Tier verwendet und bearbeitet werden.

2.1.4 EIS Tier

Der EIS Tier beinhaltet die Datenbank, und sorgt für die dauerhafte Aufbewahrung der Daten.

Im Applikationsserver GlassFish ist eine Java Datenbank, wird auch als Derby Datenbank bezeichnet, integriert. Diese integrierte Datenbank reicht für teamMgmt aus und wird deshalb auch verwendet. Dies erleichtert dem Nutzer das Aufsetzten der Serverumgebung, da er sich nicht um einen Datenbankserver und dessen Anbindung kümmern muss. Die Entwicklung wird dadurch ebenfalls erleichtert, da die Treiber für die Datenbank bereits im Server vorhanden und nicht extern eingebunden werden müssen.

3. Architektonische Ziele & Einschränkungen

3.1 Implementationsstrategie

Beim System teamMgmt wird eine Server-Client-Architektur eingesetzt. Die komplette Implementierung und die Datenbank befinden sich auf dem Server. Die Clients verwenden lediglich einen Webbrowser zur Darstellung der Antworten des Servers.

3.1.1 Daten

Um Probleme im Bereich der Synchronisation und auch im Hinblick auf die geringen Datenmengen pro Aktion, welche übertragen werden, wurde entschieden, dass die gesamte Datenhaltung auf der Serverseite stattfindet. D.h. die Clients speichern keine Daten lokal ab. Die Daten werden über Requests vom Client angefordert und anschliessend im Webbrowser dargestellt. Dies impliziert, dass eine Netzwerkverbindung zum Server zwingend nötig ist, um mit der Applikation arbeiten zu können.

3.1.2 Kommunikation

Die Kommunikation erfolgt mittels HTML Post und Get Nachrichten. Der Client fordert eine gewünschte Aktion und erhält vom Server die entsprechende Antwort, welche er dann im Webbrowser darstellt.

3.1.3 Sicherheit

Die Verwendung der Applikation und deren Funktionalitäten erfordert, dass sich der Benutzer authentifiziert. Der Ablauf dieser Authentifizierung ist wie folgt:

- Für die Anmeldung am System besitzt der Benutzer einen Benutzernamen und ein Passwort, welches er dem Server durch Eingabe übermittelt.
- Ist die Eingabe korrekt, so wird für diesen Benutzer ein Session Bean erstellt, welches den aktuell eingeloggten Benutzer inklusive seiner geladenen Rolle mit den entsprechenden Rechten enthält. Ist die Eingabe inkorrekt, so wird dies dem Benutzer angezeigt und eine erneute Eingabe ist erforderlich.
- Als weiterer Sicherheitsaspekt kommt hinzu, dass ein Benutzer nur einmal am System angemeldet sein kann (nur eine Session aktiv sein kann pro Benutzer). Eine Session wird durch den Anmeldeprozess aktiviert und durch den Abmeldeprozess oder das automatische Ablaufen der Session deaktiviert.
- Meldet sich nun ein Benutzer an, der bereits eine andere aktive Session besitzt, so wird die bisherige Session deaktiviert und die Neue aktiviert.

Dieses Session Bean, welches nach der korrekten Anmeldung eines Benutzers instanziiert wird, beinhaltet nicht nur den aktuell eingeloggten Benutzer, sondern auch seine Rechte innerhalb der Applikation. Mittels dieses Beans wird die Navigation aufgebaut und die Menüs werden entsprechend der Rechte des Benutzers geladen. Die gesamte Verwaltung der Rechte wird über dieses Session Bean gesteuert.

3.1.4 Session Handling

Wie bereits im vorherigen Abschnitt *Sicherheit* erwähnt, kann sich ein Benutzer nur einmal am System anmelden. Ein Benutzer kann also maximal eine aktive Session für die laufende Applikation besitzen. Versucht sich der Benutzer erneut einzuloggen (innerhalb einer neuen Session), so wird die alte Session invalidiert und die neue wird aktiviert.

Wenn sich ein Benutzer erfolgreich am System anmeldet, kommt folgende Codezeile zur Ausführung:

```
FacesContext.getCurrentInstance().getExternalContext().getSessionMap().put("employee", employee);
```

Diese Zeile fügt den aktuell neu eingeloggten Benutzer als Objekt der Klasse Employee der Session Map vom Faceskontext hinzu.

Parallel dazu wird folgende Codezeile ausgeführt, welche den beim Anmelden eingefügten Eintrag wieder aus der Session Map löscht, sobald sich ein Benutzer am System abmeldet:

```
FacesContext.getCurrentInstance().getExternalContext().getSessionMap().remove("employee");
```

Da die Klasse Employee das HttpSessionBindingListener Interface implementiert werden beim Hinzufügen von einem Employee Objekt in die Session Map die Methode valueBound und beim Entfernen valueUnbound automatisch ausgeführt. Ausserdem wird die valueUnbound Methode ausgeführt, sobald eine Session automatisch abläuft, somit wird der Benutzer automatisch ausgeloggt.

```
public class Employee implements HttpSessionBindingListener {
    ...

    private static Map<Integer, HttpSession> logins = new HashMap<Integer, HttpSession>();

    ...

    @Override
    public void valueBound(HttpSessionBindingEvent event) {
        HttpSession session = logins.remove(this.getId());
        if (session != null) {
            session.invalidate();
        }
        logins.put(this.getId(), event.getSession());
    }

    @Override
    public void valueUnbound(HttpSessionBindingEvent event) {
        HttpSession session = logins.remove(this.getId());
        if (session != null) {
            session.invalidate();
        }
    }
}
```

Abbildung 28: Software Architektur Dokument: Klasse Employee

Die Klasse Employee, siehe obere Abbildung, hält sich lokal eine Map mit allen eingeloggten Benutzern und deren Sessions. Wird beim Anmelden die Methode valueBound ausgeführt, so wird, sofern sich der gleiche Benutzer bereits am System angemeldet hat, die entsprechend „ältere“ Session invalidiert. Der Benutzer, der sich soeben eingeloggt hat, kann unbemerkt seine Arbeit durchführen.

Die invalidierte Session wird bei erneutem Gebrauch direkt auf den Anmeldebildschirm umschalten und ein erneutes Anmelden fordern. Dies funktioniert mittels eines PhaseListeners, der im Falle einer neuen Session vor Beginn einer Phase die Hauptseite neu lädt. Durch die Navigationskontrolle, welche neu erstellt wurde und pro Session gültig ist, ist nun wieder der Anmeldebildschirm als Inhalt für die Hauptseite gesetzt.

3.1.4.1 Lesezeichen

Fehler beim Setzen von direkten Lesezeichen sind ebenfalls nicht möglich, da es nur eine Seite für die gesamte Applikation gibt. Es wird jeweils nur der Inhalt der Hauptseite neu geladen. Wird also ein Lesezeichen gesetzt, so wird immer die Hauptseite geladen. Ist die Session noch aktiv, so wird der zuletzt benutze Bildschirm angezeigt, ansonsten der Anmeldebildschirm.

3.2 Verwendete Technologien

Die verwendeten Technologien, die bei der Entwicklung vom Produkt teamMgmt zum Einsatz kommen, werden im Folgenden aufgelistet:

- **Primefaces 3.2** - Bei Primefaces handelt es sich um ein Komponentenframework, welches die JSF Implementierungen erweitert. Es bietet spezielle User Interface Komponenten und wird in der zu Beginn des Projektes aktuellen und stabilen Version 3.2 verwendet.
- **JSF 2.0** - Java Server Faces (JSF) ist ein Framework Standard zur Entwicklung von User Interfaces für Webapplikationen. Es gehört zu den Webtechnologien der Java EE (Enterprise Edition) Plattform und wird in der aktuellen Version 2.0 verwendet, welche Ajax unterstützt.
- **EJB 3.1** - Die Enterprise Java Beans (EJB) sind standardisierte Komponenten innerhalb eines Java EE Servers und dienen zum Umsetzen wichtiger Konzepte, welche für die Geschäftslogik einer Anwendung nötig sind. Verwendet wird die aktuelle Version 3.1.
- **JPA 2.0** - Die Java Persistence API (JPA) ist eine Schnittstelle für Java Applikationen, welche die Zuordnung und die Übertragung von Objekten zu Datenbankeinträgen vereinfacht. Verwendet wird die aktuelle Version 2.0.
- **GlassFish 3** - Bei GlassFish handelt es sich um ein Open Source Java EE Server, welcher in der Version 3 verwendet wird.

4. Logische Architektur

Die Software teamMgmt ist in 3 Schichten unterteilt: User Interfaces (WebContent), Geschäftslogik (bll, Business Logic Layer) und Datenzugriffsschicht (dal, Data Access Layer). Diese drei Schichten werden in den folgenden Abschnitten genauer beschrieben.

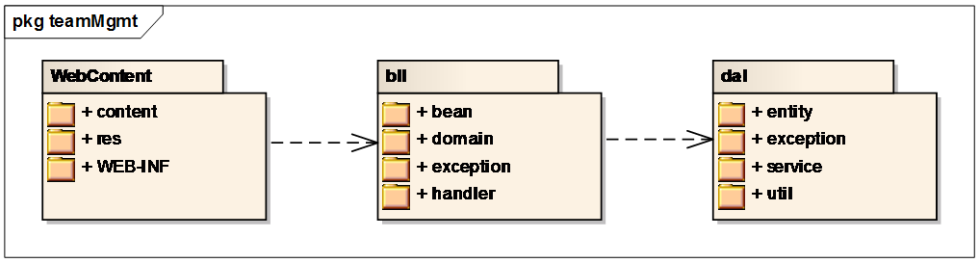

Abbildung 29: Software Architektur Dokument: Logische Architektur

4.1 User Interfaces

Die User Interfaces (WebContent) beinhalten alle für die Darstellung relevanten Dateien, Libraries, Bilder, Ressourcen und Konfigurationen.

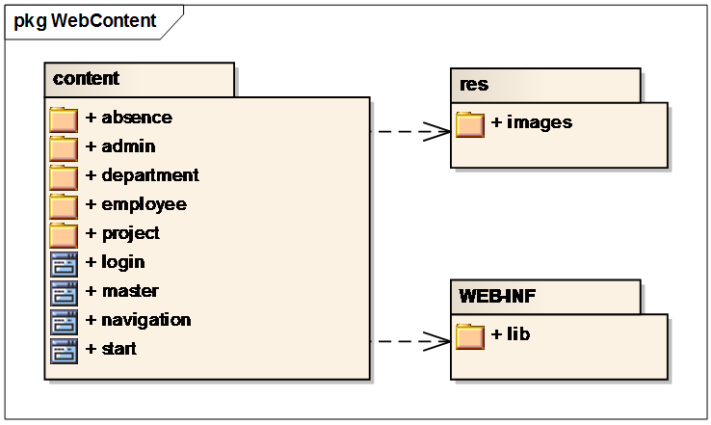

Abbildung 30: Software Architektur Dokument: WebContent

Die einzelnen Ordner innerhalb von WebContent sind in der folgenden Tabelle genauer beschrieben.

Ordner	Beschreibung
content	Alle xhtml Dateien zur Darstellung der einzelnen User Interfaces.
res	Ressourcen (wie Bilder und Stylesheet Dokument), welche zur Darstellung in den User Interfaces vom Ordern content verwendet werden.
WEB-INF	Konfigurationen die zur Beschreibung der Managed Beans dienen oder zu applikatorischen Einstellwerten. Auch die benötigen Web Libraries sind im Unterordner lib abgelegt.

Tabelle 39: Software Architektur Dokument: WebContent Ordner

4.1.1 Schnittstellen

Der WebContent (die xhtml Dateien) greifen ausschliesslich auf die Managed Beans zu, welche im Package bll.bean in der Geschäftslogik abgelegt sind.

4.1.2 Wichtige interne Abläufe

Das GUI ist in drei Bereiche aufgeteilt: Kopfzeile (mit Logo), Navigation und Inhalt. Diese drei Bereiche sind alle in die Hauptseite integriert. Bei den meisten Interaktionen des Benutzers auf dem Navigationsmenü, der einen Wechsel zu einer anderen Seite zur Folge hat, muss nur der entsprechende Inhalt neu geladen werden. Die Navigation und die Kopfzeile werden nicht nochmals übermittelt. Die folgende Abbildung zeigt den Aufbau dieser Hauptseite:

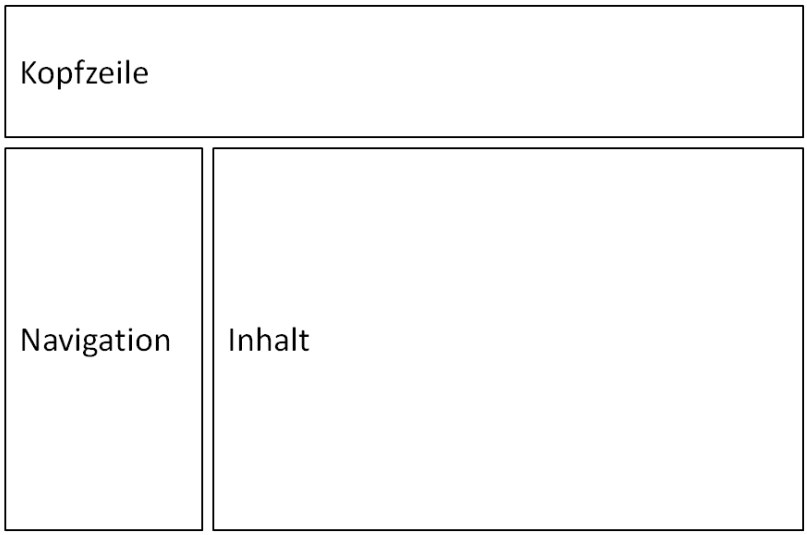

Abbildung 31: Software Architektur Dokument: Aufbau der Hauptseite

Der neu zu ladende Inhalt wird im NavigationController gesetzt und durch das erneute Laden der Hauptseite wird dann der neue Inhalt dynamisch geladen. Die Kopfzeile und die Navigation sind innerhalb der Hauptseite fest gesetzt.

Auf die Verwendung von des JSF Template Mechanismus wurde aus technologischen Gründen verzichtet. Auf dieses Problem wird im *Technischen Bericht* genauer eingegangen.

4.2 Geschäftslogik

Das Package bll beinhaltet die Geschäftslogik der Software. Es wird unterteilt in 4 weitere Packages, welche in den folgenden Abschnitten genauer beschrieben sind.

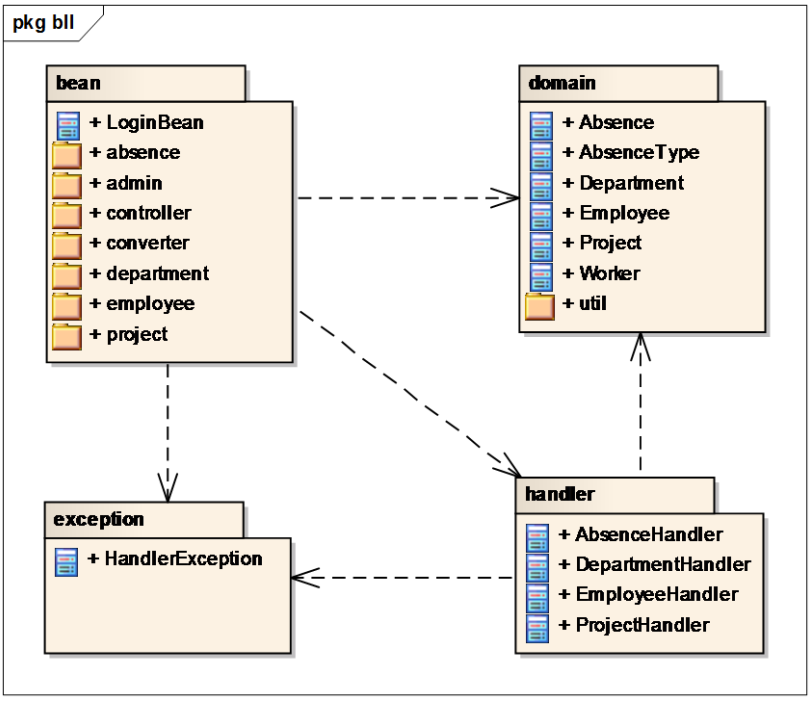

Abbildung 32: Software Architektur Dokument: Geschäftslogik

4.2.1 Subpackages

4.2.1.1 Package bean

In diesem Package sind jene Klassen untergebracht welche als Managed Beans konfiguriert sind. Diese Klassen beinhalten die von den User Interfaces verwendeten Elemente und Methoden. Ausserdem nehmen sie alle Interaktionen des Benutzers entgegen und verarbeiten diese.

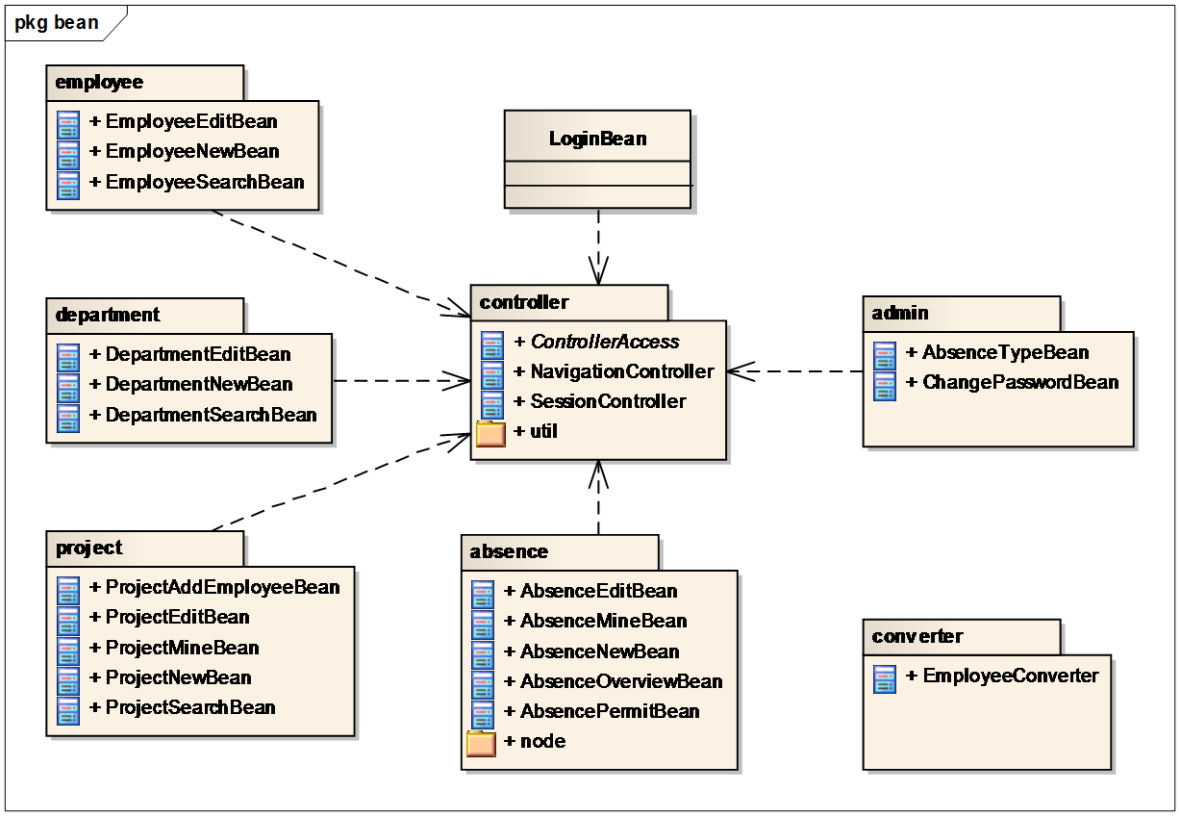

Abbildung 33: Software Architektur Dokument: Package bean

Die einzelnen Packages und Klassen innerhalb des bean Packages sind in der folgenden Tabelle genauer beschrieben.

Package / Klasse	Beschreibung
employee	Von den User Interfaces bezüglich Mitarbeiter verwendeten Managed Beans.
department	Von den User Interfaces bezüglich Abteilungen verwendeten Managed Beans.
project	Von den User Interfaces bezüglich Projekte verwendeten Managed Beans.
absence	Von den User Interfaces bezüglich Absenzen verwendeten Managed Beans. Ausserdem befinden sich die verschiedenen Arten von Nodes (node), welche für die Darstellung des Genehmigungsbaums im User Interface verwendet werden.
admin	Von den User Interfaces bezüglich Administration verwendeten Managed Beans.
controller	Controller, welche einen Benutzer während des Verlaufs einer Session unterstützen. NavigationController enthält alle Daten, welche für die Navigation verwendet werden und SessionController steuert eine aktive Sitzung des eingeloggten Benutzers. Im Package util befindet sich noch die Rolle, welche ein Benutzer einnimmt. Sie stellt alle nötigen Methoden und Funktionalitäten bezüglich dieser Rolle zur Verfügung. Auch die Klasse SessionPhaseListener, welche eine wichtige Rolle im Zusammenhang mit dem Session Handling spielt, befindet sich ebenfalls in diesem util Package.
converter	Wird verwendet zu Darstellungszwecken eines Employee Domainobjektes.
util	Beinhaltet die Klasse BeanUtil, welche einzig und alleine als Schnittstelle zu den Session Beans (SessionController und NavigationController) innerhalb des controller Packages dienen. Dies, da diese an mehreren Stellen innerhalb der Beans verwendet werden und einen komplexeren Zugriff aufweisen.
LoginBean	Diese Klasse handelt das Login eines Benutzers.

Tabelle 40: Software Architektur Dokument: bean Subpackages und Klassen

Bei allen Klassen welche auf Bean oder Controller enden handelt es sich um Managed Bean Klassen. Diese sind alle mit dem Session Scope versehen, um Fehlermeldungen direkt auf demselben Bildschirm anzeigen zu können und da für deren Anzeige andere Werte von der Datenbank geladen werden müssen. Einzig das ChangePasswordBean hat den Scope Request, da keine Daten für die Anzeige des Bildschirms geladen werden müssen.

4.2.1.2 Package domain

Dieses Package beinhaltet die Domainobjekte, welche von der Geschäftslogik benutzt werden. Innerhalb der Domainobjekte wird jeweils das entsprechende Entity Objekt gehalten.

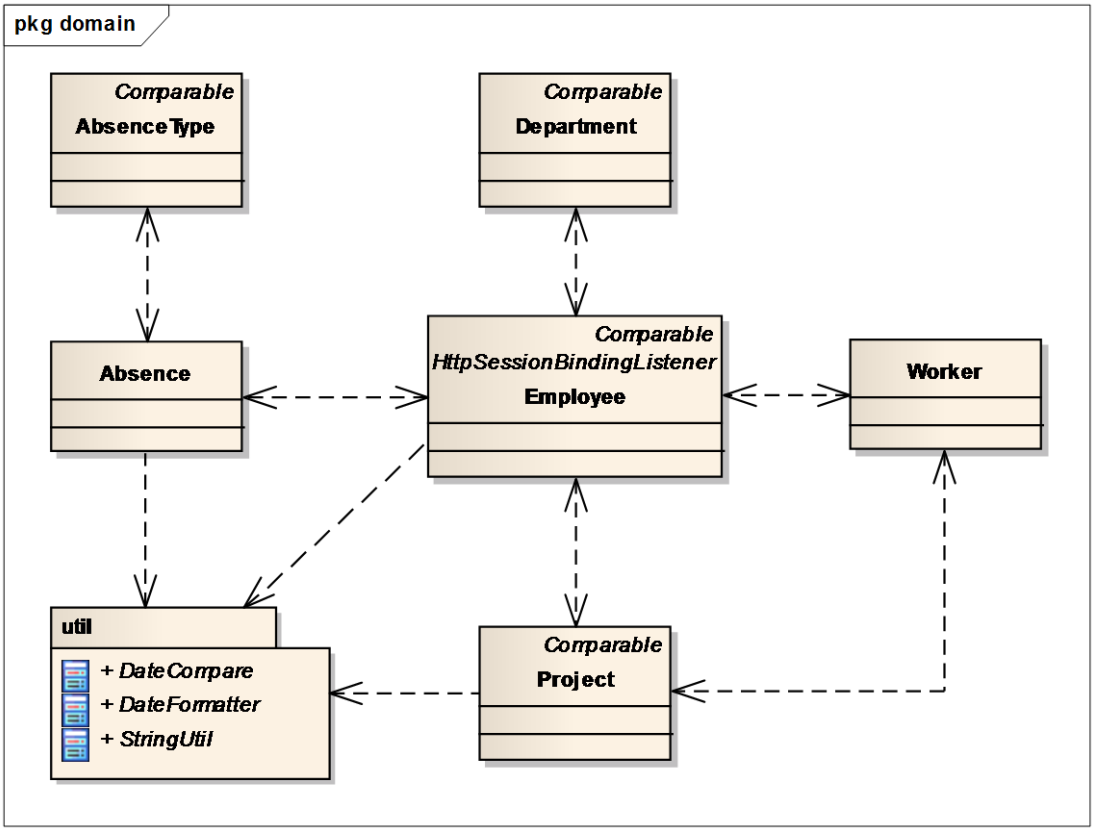

Abbildung 34: Software Architektur Dokument: Package domain

Die einzelnen Packages und Klassen innerhalb des domain Packages sind in der folgenden Tabelle genauer beschrieben.

Package / Klasse	Beschreibung
Employee	Domainklasse für einen Mitarbeiter.
Department	Domainklasse für eine Abteilung.
Project	Domainklasse für ein Projekt.
Worker	Domainklasse für einen Projektmitarbeiter.
Absence	Domainklasse für eine Absenz.
AbsenceType	Domainklasse für einen Absenztyp.
util	Bietet unterstützende Klassen für die einzelnen Domainklassen an. DateCompare dient zum Vergleich von Daten. DateFormatter formatiert Daten zur gewünschten Darstellung in den einzelnen User Interfaces. StringUtil bietet den Vergleich von Strings an (ob null oder nicht).

Tabelle 41: Software Architektur Dokument: domain Subpackages und Klassen

4.2.1.3 Package handler

In diesem Package befinden sich die Handler, welche als Schnittstelle zur Datenzugriffsschicht dienen. Die Handler sind in vier Klassen gemäss den vier Einsatzbereichen unterteilt und jeweils als Singleton implementiert. Die einzelnen Methoden werden von den Managed Beans verwendet.

Ebenfalls in dieser Klasse erfolgt eine Umwandlung der Entity Objekte (dienen zur Speicherung der Daten auf der Datenbank) in Domainobjekte, welche innerhalb der Geschäftslogik verwendet werden.

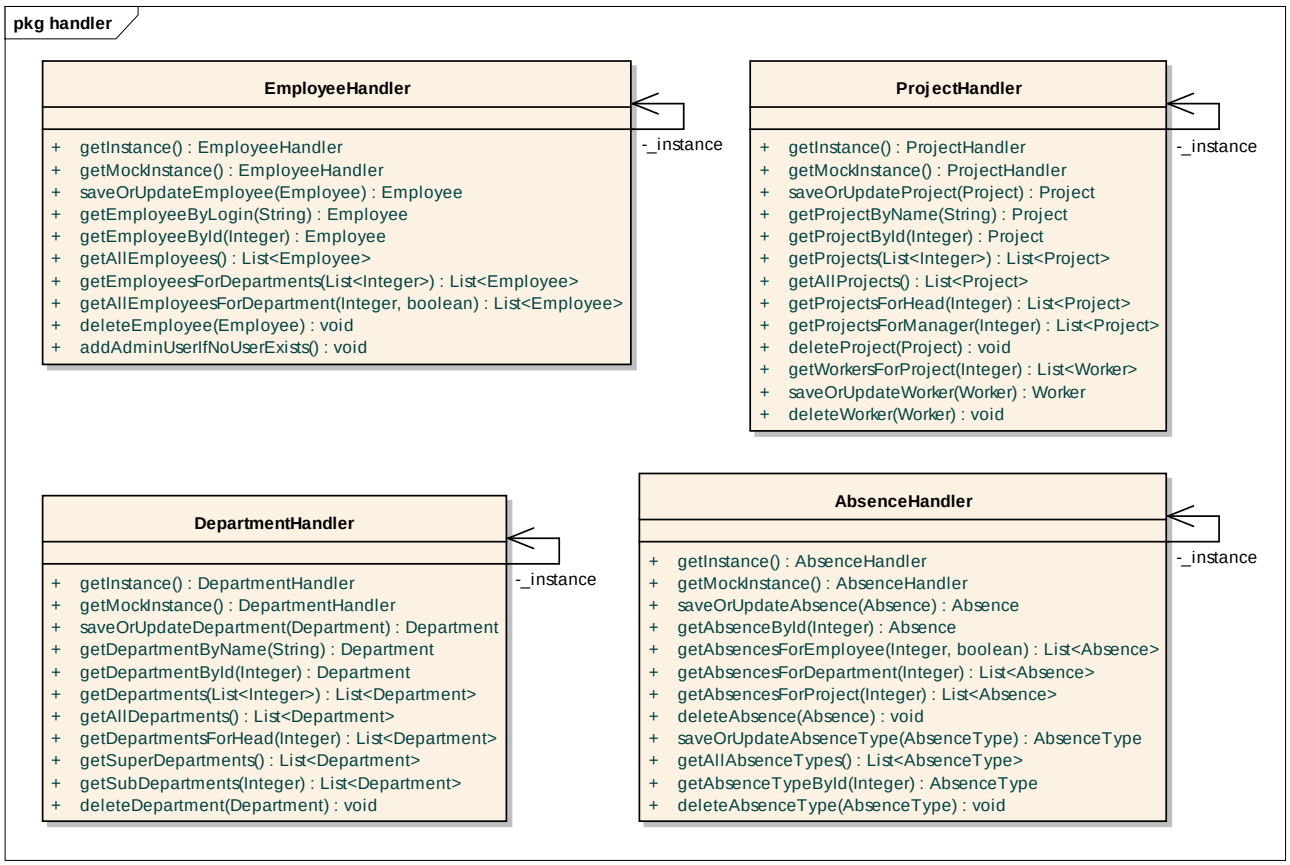

Abbildung 35: Software Architektur Dokument: Package handler

Die einzelnen Klassen innerhalb des handler Packages sind in der folgenden Tabelle genauer beschrieben.

Klasse	Beschreibung
EmployeeHandler	Bietet alle Methoden im Zusammenhang mit Mitarbeitern an.
DepartmentHandler	Bietet alle Methoden im Zusammenhang mit Abteilungen an.
ProjectHandler	Bietet alle Methoden im Zusammenhang mit Projekten an.
AbsenceHandler	Bietet alle Methoden im Zusammenhang mit Absenzen an.

Tabelle 42: Software Architektur Dokument: handler Klassen

4.2.1.4 Package exception

In diesem Package befindet sich nur die Klasse HandlerException, welche von den einzelnen Handlern (im Abschnitt *Package handler* beschrieben sind) im Falle eines Datenbankfehlers geworfen werden. Diese Exception wird dann in den Beans entsprechend verarbeitet um in den User Interfaces angezeigt werden zu können. Innerhalb der HandlerException Klasse wird die entsprechende Fehlermeldung gespeichert.

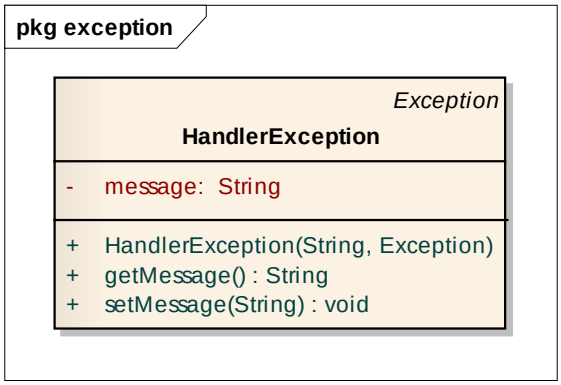

Abbildung 36: Software Architektur Dokument: Package exception

4.2.2 Schnittstellen

Die Managed Beans des bean Packages werden von den User Interfaces des WebContents verwendet und behandeln jede Interaktion eines Benutzers mit dem entsprechenden User Interface. Pro User Interface existiert eine entsprechende Managed Bean Klasse.

Die Handler Klassen des handler Packages stellen alle nötigen Methoden zur Ansicht oder Manipulation von Daten zur Verfügung, welche von den Managed Beans benötigt werden.

4.2.3 Wichtige interne Abläufe

Die Methoden der Handler Klassen im handler Package werden ausschliesslich von den Managed Beans des Packages bean benutzt. Sie dienen als Schnittstelle zwischen der Geschäftslogik und dem effektiven Datenzugriff auf der Datenbank. Ausserdem werden Entity Objekte der Datenzugriffsschicht in Domainobjekte der Geschäftslogik umgewandelt und umgekehrt.

4.3 Datenzugriffsschicht

Das Package dal beinhaltet alle nötigen Entities, welche auf der Datenbank gespeichert werden, sowie die Serviceschnittstelle, die den Zugriff auf die Datenbank steuert. Das Package selbst ist unterteilt in vier weitere Packages, welche in den folgenden Abschnitten genauer beschrieben sind.

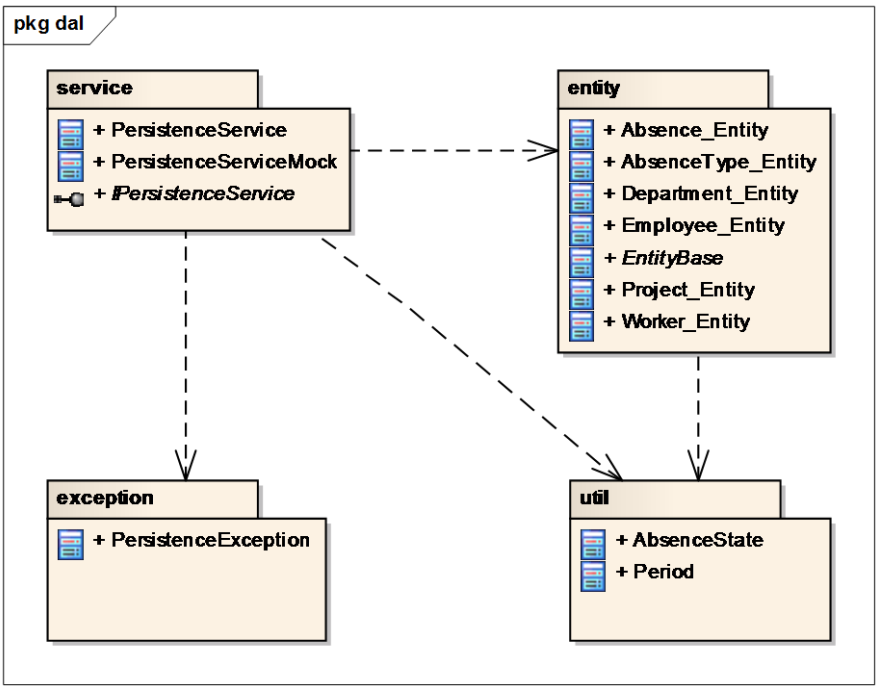

Abbildung 37: Software Architektur Dokument: Datenzugriffsschicht

4.3.1 Subpackages

4.3.1.1 Package service

Im service Package befindet sich die effektive Schnittstelle zur Datenbank.

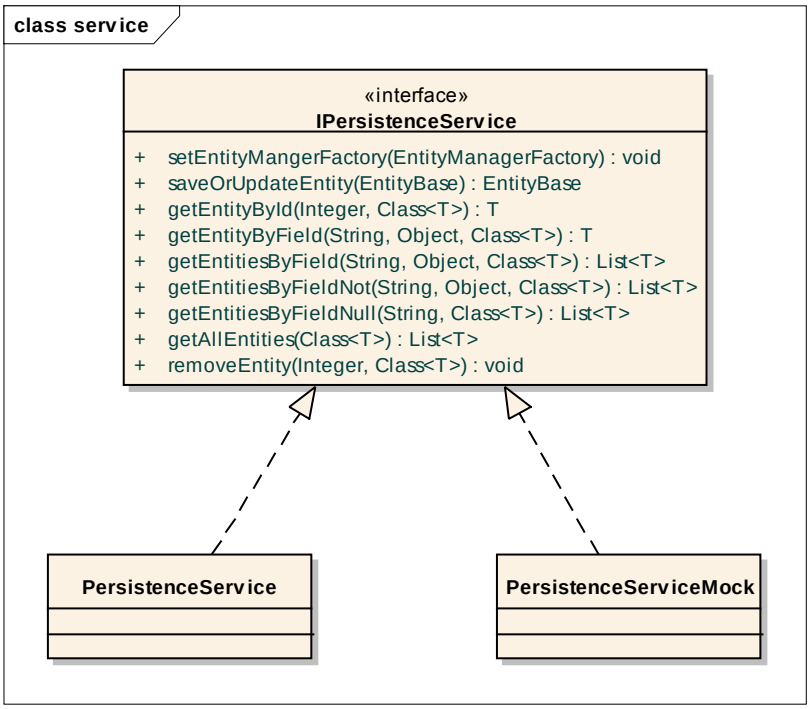

Abbildung 38: Software Architektur Dokument: Package service

Die einzelnen Klassen innerhalb des util Packages sind in der folgenden Tabelle genauer beschrieben.

Klasse	Beschreibung
IPersistenceService	Interface welches von der Geschäftslogik verwendet wird.
PersistenceService	Implementierung des Service mit Zugriff auf die Datenbank.
PersistenceServiceMock	Mock Service, welcher einen Datenbankzugriff simuliert und ausschliesslich mit lokalen Daten arbeitet.

Tabelle 43: Software Architektur Dokument: service Klassen

4.3.1.2 Package entity

Dieses Package enthält alle Entities, welche auf der Datenbank abgelegt werden. Bei den Entities handelt es sich um praktisch reine Datenklassen, welche wenig bis keine Logik enthalten. Es handelt sich um Enterprise Java Bean (EJB) Klassen.

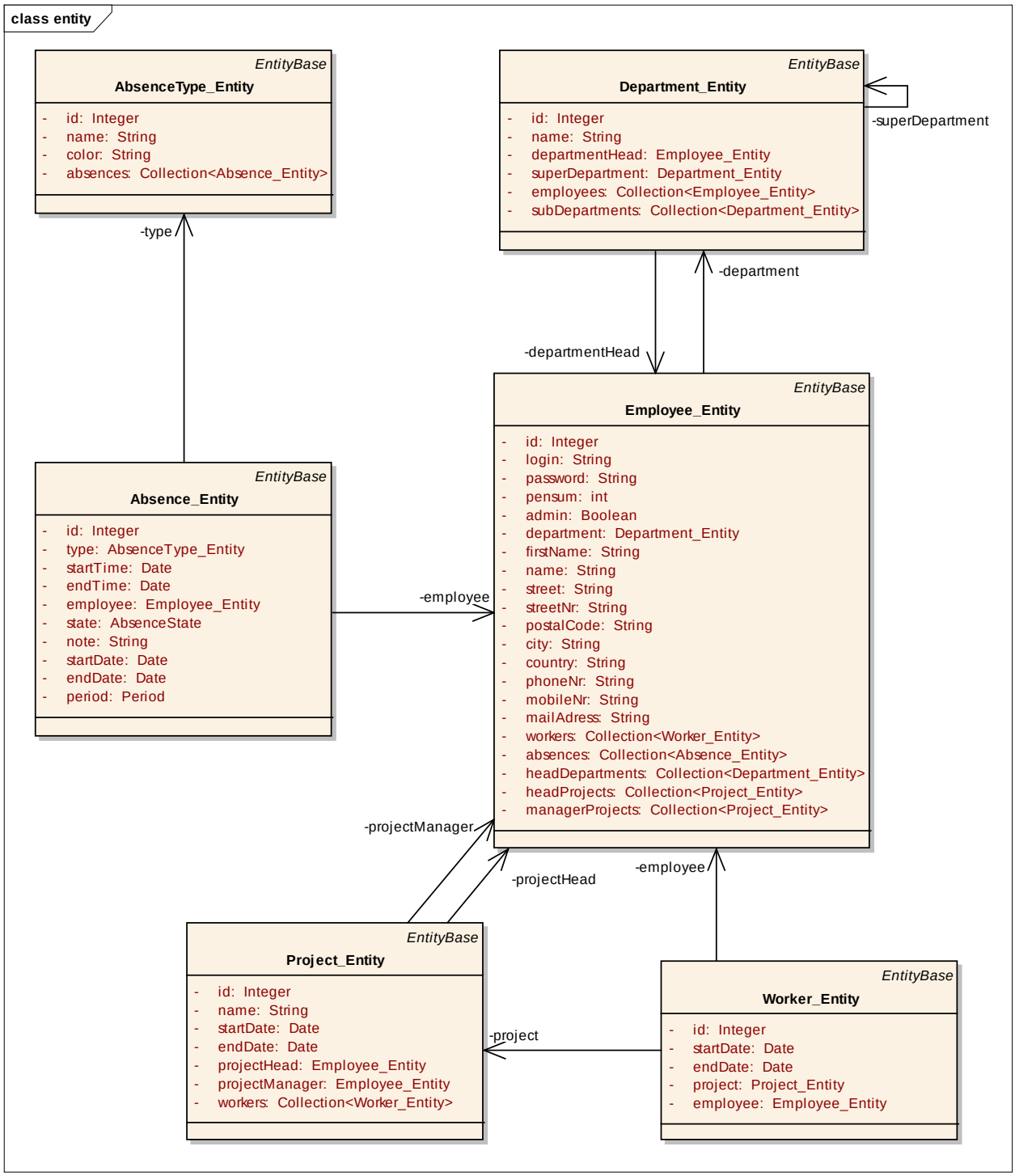

Abbildung 39: Software Architektur Dokument: Package entity

Alle Entity Klassen erben von der EntityBase Klasse, welche in der folgenden Abbildung ersichtlich ist.

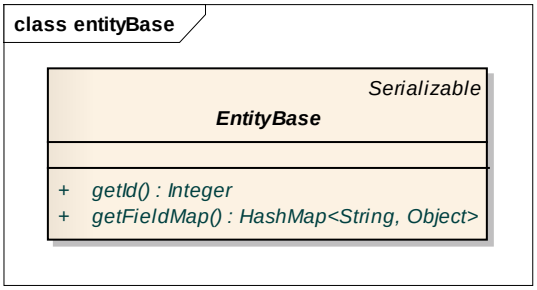

Abbildung 40: Software Architektur Dokument: Klasse EntityBase

Diese EntityBase Klasse fordert die Implementierung der Methode getId, die jede Entity Klasse haben muss um Entity Klassen dynamisch verwenden zu können. Ausserdem wird die Methode getFieldMap gefordert, welche für die Suche unter Verwendung der „gemockten“ Daten der Entities benötigt wird.

4.3.1.3 Package exception

Die PersistenceException vereinheitlicht die verschiedenen Fehlerfälle die bei der Persistierung auftreten können. Der PersistenceService fängt alle Exceptions des JPA Containers ab und wirft eine PersistenceException, welche dann innerhalb der Handler Klassen des bll.handler Packages abgefangen und in HandlerExceptions umgewandelt werden. Diese wird wiederum in den Bean Klassen des bll.bean Packages abgefangen und informiert danach den Benutzer über den aufgetretenen Fehler.

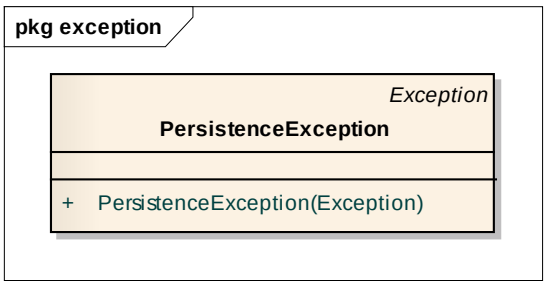

Abbildung 41: Software Architektur Dokument: Package exception

4.3.1.4 Package util

Dieses Package beinhaltet Hilfsklassen die innerhalb des dal Packages verwendet werden.

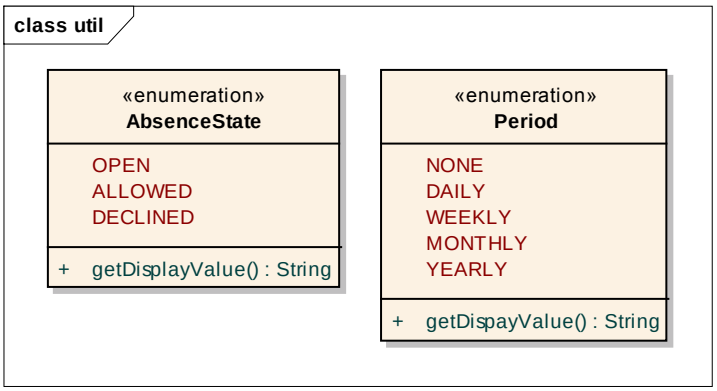

Abbildung 42: Software Architektur Dokument: Package util

Die einzelnen Klassen innerhalb des util Packages sind in der folgenden Tabelle genauer beschrieben.

Klasse	Beschreibung
Period	Enumeration, welche die Art der Periodizität beinhaltet. Wird von der Klasse Absence_Entity verwendet.
AbsenceState	Enumeration, welche den Status einer Absenz beinhaltet. Wir ebenfalls von der Klasse Absence_Entity verwendet.

Tabelle 44: Software Architektur Dokument: util Klassen

4.3.2 Schnittstellen

Das PersistenceService Interface wird ausschliesslich von den Handler Klassen des bll.handler Packages verwendet.

4.4 Wichtige Abläufe

Einer der wichtigsten Abläufe, welcher packageübergreifend stattfindet, ist die Verbindung der User Interfaces mit den Daten der Datenbank. Die User Interfaces (xhtml Seiten innerhalb des content Packages) haben Zugriff auf die Managed Beans (bean Package) und die entsprechenden Domainklassen (domain Package). Die Managed Beans greifen über die Handler Klassen des handler Packages auf das IPersistenceService Interface und somit auf die Datenbank zu. Die Handler Klassen wandeln zudem datenbankspezifische Entities (entity Package) und Exceptions (PersistenceException) in Domainklassen und HandlerExceptions um. Eine Domainklasse beinhaltet immer das zugehörige Entity und bietet Methoden für den Zugriff auf die gewünschten Werte an.

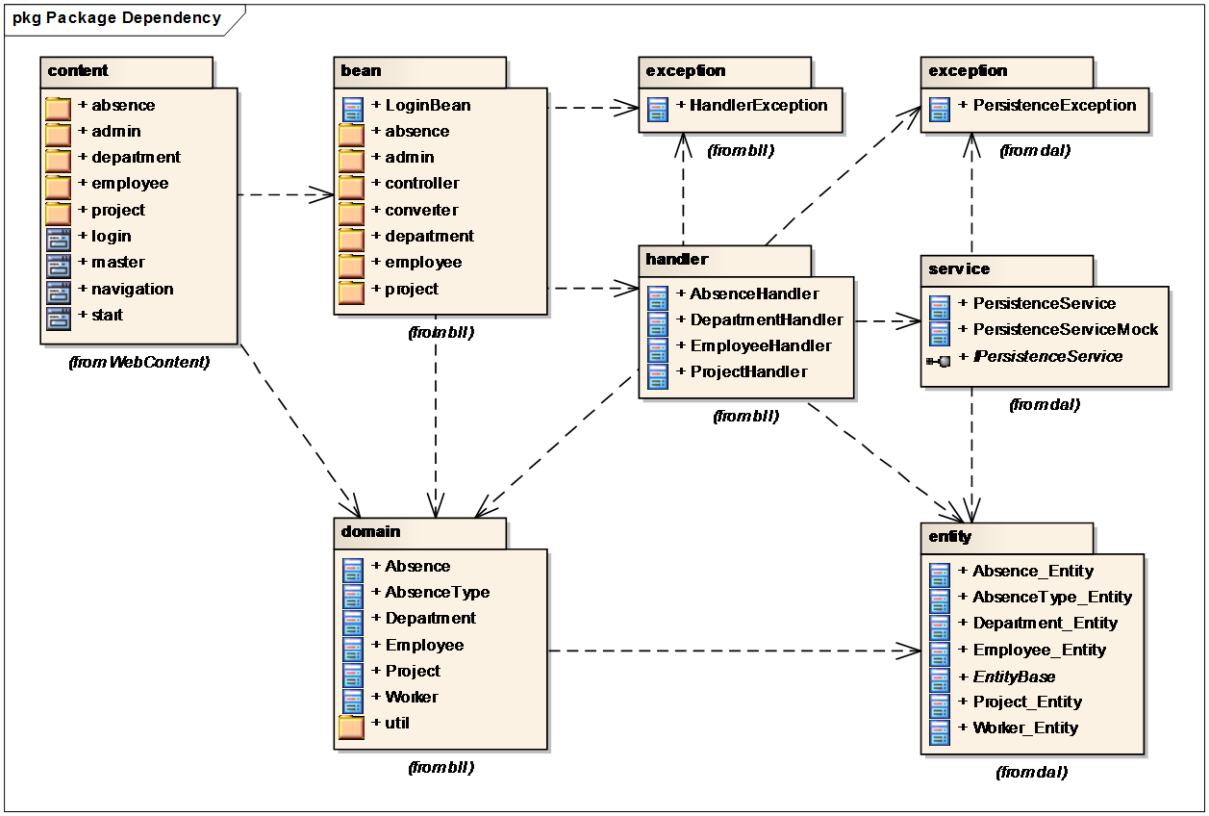

Abbildung 43: Software Architektur Dokument: Package übergreifende Abhängigkeiten

Dieser Ablauf wird in folgendem Beispiel innerhalb eines Systemsequenzdiagrammes gezeigt. Es handelt sich um ein Beispiel wo ein Benutzer den Bildschirm zum erstellen eines neuen Mitarbeiters aufruft und dafür die nötigen Abteilungen geladen werden müssen, welche einem Benutzer hinzugefügt werden können.

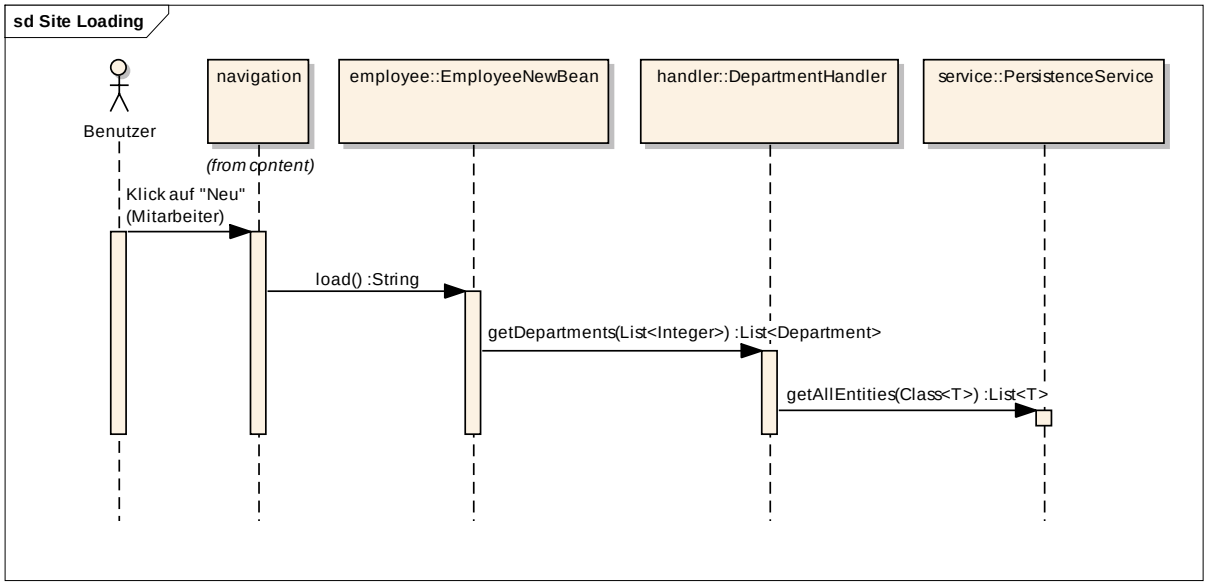

Abbildung 44: Software Architektur Dokument: Sequenzdiagramm

5. Prozesse und Threads

Auf dem GlassFish Server läuft die Applikation innerhalb eines Prozesses (Service). Beim Start dieses Prozesses wird auch der Datenbankprozess gestartet.

Pro Benutzer, welcher sich am System anmeldet, stellt der GlassFish Server automatisch einen entsprechenden Thread zur Verfügung. Somit können mehrere Threads innerhalb des Systems laufen, da sich mehrere Benutzer gleichzeitig am System anmelden können. Falls es sich um Zugriffe (Schreibzugriffe) auf gemeinsam verwendete Daten handelt werden diese synchronisiert. Aus Gründen der Performance sind nur schreibende Zugriffe synchronisiert. Durch die Synchronisation der Schreibzugriffe auf die Datenbank, kann verhindert werden, dass undefinierte, korrupte Datensätze entstehen. Es kann jedoch nicht ausgeschlossen werden, dass ein Benutzer auf Grund veralteter Daten ein ungültiges Objekt besitzt. Wenn zwei oder mehrere gleichzeitig ein Objekt bearbeiten wird die Speicherung des Zweiten die Änderungen des Ersten überschreiben. Es gilt das Prinzip, der Letzte gewinnt. Dieser Kompromiss wird bewusst eingegangen um den Datentransfer vom Server zu den Benutzern zu minimieren und da die stetige Aktualität der Daten nicht von hoher Wichtigkeit ist. Bei einem erneuten Laden oder Seitenwechsel findet diese Aktualisierung der Daten statt.

6. Deployment

Das Produkt wird auf einen GlassFish Applikationsserver geladen. Als Datenbank wird die mit dem GlassFish Server mitgelieferte Java Datenbank (Derby) verwendet. Dieser Service muss lediglich gestartet werden. Eine entsprechende Anleitung zum laden und verwalten der Applikation und des Servers wird mitgeliefert.

7. Datenspeicherung

Die Datenhaltung für das Produkt teamMgmt erfolgt mittels JPA und einer Java Datenbank (Derby), welche im Applikationsserver GlassFish integriert ist.

Die Daten werden nur auf der Serverseite gehalten, das heisst die Clients speichern domainrelevante Daten niemals lokal ab. Dies verhindert Synchronisationsprobleme und stellt sicher, dass zu einem Zeitpunkt jeweils nur eine gültige Version der Daten existiert.

Die folgende Abbildung zeigt das Datenmodell der Datenbank im Detail.

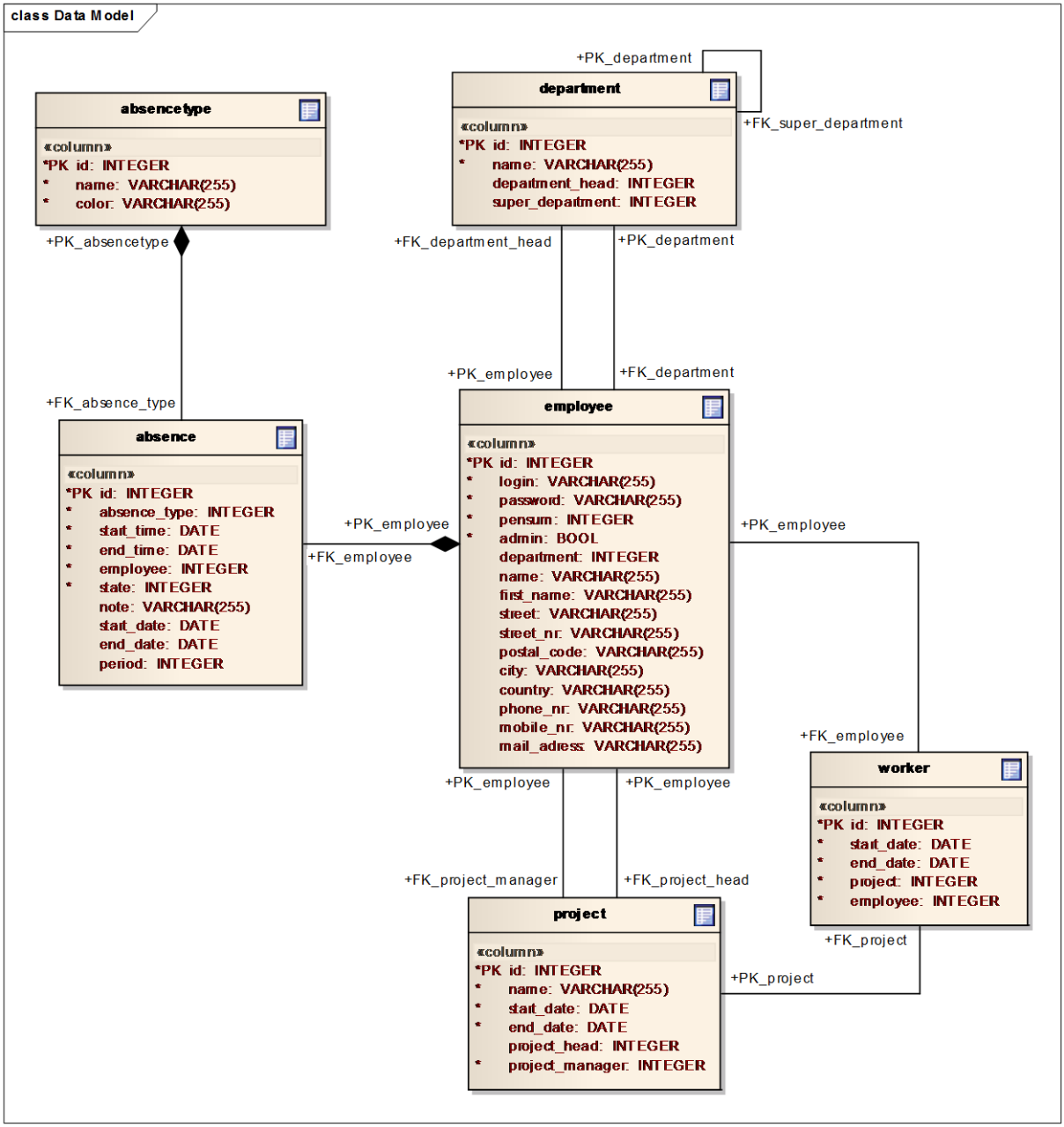

Abbildung 45: Software Architektur Dokument: Datenmodell

8. Grössen und Leistung

8.1 Mitarbeiter

Bei der Auflistung von Mitarbeitern wird ein Paging Mechanismus verwendet. Sobald die Liste mehr als 10 Einträge enthält wird dieser Mechanismus angewendet. Ansonsten werden die Daten ohne Paging Overhead angezeigt.

8.2 Projekte

Die Auflistung der Projekte beinhaltet nur Projekte, welche noch nicht beendet wurden. Ausserdem wird auch hier derselbe Paging Mechanismus wie bei den Mitarbeitern angewendet, was bedeutet, dass ab einer Menge von 10 Einträgen innerhalb der Liste die Liste mit Paging Mechanismus dargestellt wird.

8.3 Absenzen

Die Anzahl der Absenzen, welche ein Mitarbeiter erfassen kann ist nicht eingeschränkt. Deshalb werden in der Ansicht jeweils nur Absenzen, welche nicht älter als zwei Monate sind geladen. Ausserdem wird bei der Auflistung der Absenzen ein Paging Mechanismus verwendet, damit das User Interface nicht unnötig gross wird.

Handbuch (Inbetriebnahme / Betrieb)

1. Inbetriebnahme

Eine komplette Hilfestellung zur Installation und Inbetriebnahme eines GlassFish Servers findet sich unter:

http://glassfish.java.net/docs/3.1/installation-guide.pdf

In diesem Dokument wird von einer lauffähigen Installation eines GlassFish Servers ab Version 3 oder höher ausgegangen.

1.1.1 Starten der Servern

Bevor Sie das Produkt teamMgmt auf dem Server installieren können müssen auf diesem folgende Schritte ausgeführt werden (Beschreibung im angegeben Kapitel):

- *2.1.2 Starten der Domäne*
- *2.1.5 Starten des Java Datenbank Servers*
- *2.1.7 Remote Administration*

1.1.2 Installation von teamMgmt

Öffnen Sie hierzu ihren Webbrowser und verbinden Sie sich mit der Administrationskonsole gemäss dem Punkt *2.1.8 Zugriff Administrationskonsole.*

2. Betrieb

2.1 Server

In den Abschnitten *2.1.1* bis *2.1.8* werden kurz die wichtigsten Arbeitsschritte am Server beschrieben. Dabei handelt sich um Auszüge aus der GlassFish Quickstart Guide. Die komplette Dokumentation findet sich hier: http://glassfish.java.net/downloads/quickstart/index.html

Anmerkung:

- *as-install* dient als Platzhalter für den Pfad zu der Installation, z.B.: */home/user/glassfish/glassfish3/*
- *as-install-parent* dient als Platzhalter für den Pfad zu der Installation, z.B.: */home/user/glassfish/*
- *domain-dir* dient als Platzhalter für den Pfad zur Domain z.B.: */home/user/glassfish3/ domains/*
- Die Kommandos müssen mit dem für ihr Betriebssystem entsprechenden Operand ausgeführt werden.

2.1.1 Standardeinstellungen

Bezeichnung	Wert	Kommentar
Domainname	domain1	
Master password	changeit	Sofern kein Anderes bei der Installation angegeben
asadmin command-line utility	*as-install*/bin	Kommandozeile des GlassFish Servers
Configuration files	*domain-dir*/config	
Log files	*domain-dir*/logs	
Administration server port	4848	Zur remote Benutzung muss Secure … eingerichtet werden. (Verwendet HTTPS)
HTTP port	8080	
HTTPS port	8181	

Tabelle 45: Handbuch: Standardeinstellungen

2.1.2 Starten der Domäne

Nach der Erfolgreichen Installation der GlassFish Server Software kann die Domäne gestartet werden.

*as-install***/bin/asadmin start-domain**

Dieses Kommando startet die Default-Domäne, *domain1*.

2.1.3 Stoppen der Domäne

Um eine Domäne zu stoppen verwenden Sie folgendes Kommando.

*as-install***/bin/asadmin stop-domain**

2.1.4 Überprüfung laufender Domänen

Um alle laufenden Domänen anzuzeigen verwenden Sie folgendes Kommando:

*as-install***/bin/asadmin list-domains**

2.1.5 Starten des Java Datenbank Servers

Um den Server der Java Datenbank zu starten verwenden Sie folgendes Kommando:

*as-install***/bin/asadmin start-database --dbhome** *as-install-parent***/javadb**

2.1.6 Stoppen des Java Datenbank Servers

Um den Server der Java Datenbank zu stoppen verwenden Sie folgendes Kommando:

*as-install***/bin/asadmin stop-database**

2.1.7 Remote Administration aktivieren

Bevor Sie diesen Schritt durch führen müssen alle Domains gestoppt sein, folgen Sie dazu den Anweisungen von Punkt *2.1.3 Stoppen der Domäne*.

Um die remote Administration zu aktivieren verwenden Sie folgendes Kommando:

*as-install***/bin/asadmin enable-secure-admin**

Nach der Ausführung des Kommandos können die Domänen wieder gestartet werden, folgen Sie dazu den Anweisungen von Punkt *2.1.2 Starten der Domäne*.

2.1.8 Zugriff Administrationskonsole

Um auf die Administrationskonsole zuzugreifen verwenden Sie Ihren Webbrowser und verbinden sich mit ihrem Server via Port: 4848:

http://ihreserveraddresse:4848

Projektmanagement

1. Projektplan

1.1 Einführung

1.1.1 Zweck

Dieses Dokument soll einen Überblick über die Planung und die Durchführung Bachelorarbeit teamMgmt (Verwaltung von Mitarbeitern und Abwesenheiten) verschaffen.

1.1.2 Gültigkeitsbereich

Die Gültigkeit dieses Projektplans erstreckt sich über das Frühjahrssemester 2012 im Rahmen einer Bachelorarbeit bis zur definitiven Abgabe am 15. Juni 2012.

1.1.3 Referenzen

Nachfolgend werden die Dokumente aufgeführt, welche von diesem Dokument referenziert werden.

- Aufgabenstellung.pdf (vom Beteuer)
- TechnischeRisiken.pdf
- Glossar.pdf

1.1.4 Übersicht

Im Kapitel *Projekt Übersicht* wird eine Übersicht über das Projekt teamMgmt gewährt. Der Zweck, die Ziele und unsere persönliche Motivation sollen ersichtlich werden.

Die einzelnen Mitarbeiter des Projektes mit ihren Aufgaben und Verantwortungen werden im Kapitel *Projektorganisation* beschrieben.

Wichtige Bereiche dieses Dokumentes sind die Kapitel *Management Abläufe* und *Arbeitspakete*. Einerseits werden dort der Projektplan, der Zeitplan, die Iterationsplanung und die Meilensteine festgelegt und andererseits wird die Handhabung von Arbeitspaketen erläutert.

Im Kapitel *Risikomanagement* erörtern wir die möglichen Risiken in unserem Projekt und schätzen den möglichen Schaden ein. Des Weiteren werden die Infrastruktur und Qualitätsmassnahmen des Projektes in den entsprechenden Kapiteln *Infrastruktur* und *Qualitätsmassnahmen* beschrieben.

1.2 Projekt Übersicht

Die Web Applikation teamMgmt soll es einem Unternehmen ermöglichen seine Mitarbeiter und deren Abwesenheiten einfach und übersichtlich zu verwalten. Mitarbeiter werden eingeteilt in Abteilungen und Projekte, wodurch gezielte Auswertungen möglich sind.

Die entstehende Applikation soll zudem als Beispielapplikation im Modul Internettechnologien an der HSR dienen. Deshalb werden auch ein Erfahrungsbericht und eine Beurteilung der eingesetzten Technologien und Werkzeuge erstellt.

1.2.1 Zweck und Ziel

Das Ziel dieser Arbeit ist es einem kleinen bis mittelgrossen Unternehmen die Verwaltung der Mitarbeiter und deren Abwesenheiten zu erleichtern. Ausserdem soll es möglich sein diverse spezifische Auswertungen dieser Abwesenheiten zu generieren um sich den benötigten Überblick verschaffen zu können. Von den einzelnen Mitarbeitern werden Kontaktinformationen (Name, Telefonnummern, Mailadressen, usw.) hinterlegt, um im Notfall diesen kontaktieren zu können.

Die antreibende Idee dahinter ist, dass es bis heute noch viele Unternehmen gibt, welche diese Arbeit mittels einer zentral abgelegten Excel Tabelle oder Ähnlichem lösen. Diese bieten meist keinen kompakten Überblick zum Beispiel für einen Projektleiter, da er sich die einzelnen Mitarbeiter seines Teams über die gesamte Tabelle zusammensuchen muss. Solche und auch noch weitere Probleme werden durch teamMgmt behoben.

Inwiefern die Resultate aus diesem Projekt für kommerzielle Zwecke verwendet werden ist noch abzuklären.

1.2.2 Lieferumfang

Folgende Punkte sind im Lieferumfang enthalten:

- Web-Applikation teamMgmt zur Verwaltung von Mitarbeitern und Abwesenheiten
- Benutzerhandbuch zur Verwendung der Applikation
- Erfahrungsbericht und Beurteilung der eingesetzten Technologien und Werkzeuge
- Vollständige Software Engineering Dokumentation

Weitere Details zum Lieferumfang sind in den Anforderungen spezifiziert.

Vordefinierte Abgaben für eine Bachelorarbeit (Dokumentation, Berichte, Poster) können den Anleitungen der HSR entnommen werden.

1.3 Projektorganisation

1.3.1 Organisationsstruktur

Name	Aufgaben	Verantwortung
Beat Bodenmann	- Entwicklung Java - Überwachung gesamte Projektdokumentation und Qualität (inkl. Abgaben & Poster) - Setup und Überwachung virtueller Server	- Protokolle erstellen - Überwachung der Qualität der Dokumente - Management des virtuellen Servers - Systemtest: Dokumentation und Durchführung
Christian Zurbuchen	- Entwicklung Java - Überwachung gesamte Projektdokumentation und Qualität (inkl. Abgaben & Poster) - Überwachung Projektplan und Zeiterfassung - Sitzungsvorbereitung	- Überwachung Qualität des Codes - Überwachung des Zeitplans - Planung des Projektes (Arbeitspakete zu den einzelnen Iterationen/Phasen) - Koordination des Projektes / Teams

Tabelle 46: Projektplan: Organisationsstruktur

1.3.2 Externe Schnittstellen

Betreuer	Prof. Hans Rudin (hrudin@hsr.ch)
Experte	Daniel Hiltebrand, Crealogix

Tabelle 47: Projektplan: Externe Schnittstellen

1.4 Management Abläufe

1.4.1 Kostenvoranschlag

Für das Projekt sind 16 Wochen geplant. Nach 15 Wochen sind die Hauptaufgaben des Projektes jedoch beendet und in der letzten Woche wird der Projektabschluss vorbereitet und letzte Abschlussarbeiten werden noch erledigt.

Das Projekt läuft in der Zeitspanne vom 20. Februar 2012 bis zum 15. Juni 2012, wobei innerhalb dieser 17 Wochen insgesamt eine Woche an Feiertagen enthalten ist.

Die einzelnen Projektmitarbeiter werden gemäss ihren Vorkenntnissen laufend den neu definierten Arbeitspaketen, jeweils zu Beginn einer neuen Iteration, zugeteilt. Wir rechnen mit einem Arbeitsaufwand von mindestens 22.5 Stunden (30 * 12 = 360 / 16) pro Woche und pro Projektmitarbeiter.

1.4.2 Zeitliche Planung

In diesem Abschnitt wird die zeitliche Planung des Projektes gemacht. Die Planung und Einteilung der Phasen und Iterationen, sowie die Festlegung der einzelnen Meilensteine.

Die folgende Abbildung soll einen Überblick über die Phasen / Iterationen geben. Die Meilensteine sind als rote Punkte direkt in der Planung markiert.

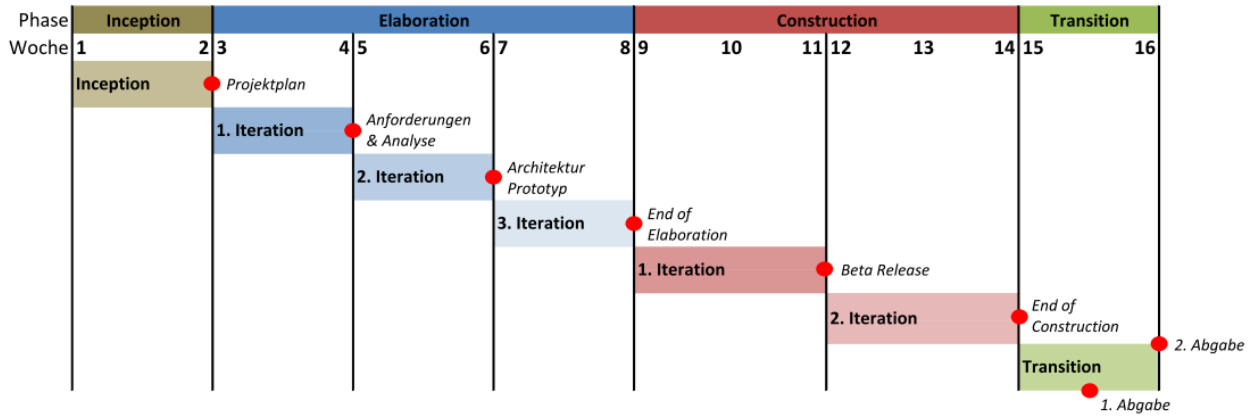

Abbildung 46: Projektplan: Überblick Phasen, Iterationen und Meilensteine

Zwischen die Projektwochen 7 und 8 fällt das Osterwochenende, weshalb eine Woche Ferien eingeplant wurde. Auf die anderen Feiertage wurde keine Rücksicht genommen.

1.4.2.1 Phasen / Iterationen

Das Projekt dauert total 16 Wochen. Diese 16 Wochen werden auf die in den folgenden Abschnitten beschriebenen Phasen aufgeteilt.

Die einzelnen Iterationen der Phasen werden mittels Iterationsplänen im Detail geplant und nach Beendigung durch ein Assessment ausgewertet.

1.4.2.1.1 Inception

Dauer: 2 Wochen

In der Phase Inception geht es darum das Projekt zu initiieren und den Projektplan festzulegen. Auch werden schon erste Anforderungen analysiert und beschrieben. Zudem werden erste Technologie- / Containerstudien betrieben und die Tauglichkeit der einzelnen Varianten erörtert.

1.4.2.1.2 Elaboration

Dauer: 6 Wochen

Die Elaboration Phase beinhaltet die Durchführung von Analyse und Design. Ausserdem werden ein Architektur und ein GUI Prototyp im Verlaufe der Phase erstellt. Die Phase wird in drei Iterationen aufgeteilt, welche in den folgenden Abschnitten genauer beschrieben werden.

1.4.2.1.2.1 1. Iteration

Dauer: 2 Wochen

In der ersten Iteration werden alle Anforderungen spezifiziert und die Analyse der Problemdomain wird durchgeführt. Die Technologie- / Containerstudien werden fortgesetzt und eine entsprechende Variante wird festgelegt.

1.4.2.1.2.2 2. Iteration

Dauer: 2 Wochen

Auf Grund der Analyse wird mit dem Design der Applikation fortgefahren (Erstellung Software Architektur Dokument, SAD). Ein erster minimaler Architektur Prototyp der Web Applikation wird erstellt.

1.4.2.1.2.3 3. Iteration

Dauer: 2 Wochen

In der letzten Iteration der Phase Elaboration wird das Design der Applikation abgeschlossen (SAD fertiggestellt) und ein erster GUI Prototyp der Web Applikation wird erstellt.

1.4.2.1.3 Construction

Dauer: 6 Wochen

Die Construction Phase beinhaltet die Implementierung der Software und Fertigstellung des Designs. Zudem werden die Erfahrungsberichte erstellt und die Dokumentation und das Benutzerhandbuch abgeschlossen. Diese Phase wird in zwei Iterationen aufgeteilt, welche in den folgenden Abschnitten genauer beschrieben werden.

1.4.2.1.3.1 1. Iteration

Dauer: 3 Wochen

Ein Beta Release der Applikation und die Architektur der Software (SAD) werden fertiggestellt. Erste Arbeiten am Benutzerhandbuch und an den Erfahrungsberichten mit JSF Werkzeugen.

1.4.2.1.3.2 2. Iteration

Dauer: 3 Wochen

Fertigstellung des Final Release der Applikation, dem Benutzerhandbuch, der Dokumentation des Projektes und an den Erfahrungsberichten mit JSF Werkzeugen.

1.4.2.1.4 Transition

Dauer: 2 Wochen

Allfällige Anpassungen werden noch gemacht und die Abgaben für die Bachelorarbeit werden abgeschlossen.

1.4.2.2 Meilensteine

In den folgenden Abschnitten werden die Meilensteine des Projektes beschrieben.

1.4.2.2.1 Projektplan

Termin: 4. März 2012

Der Aufbau des Projektes ist abgeschlossen und das Projektplandokument ist überarbeitet und fertiggestellt.

1.4.2.2.2 Anforderungen & Analyse

Termin: 18. März 2012

Alle Anforderungen sind spezifiziert und die Analyse der Problemdomain ist abgeschlossen. Technologie und Container wurden ausgewählt.

1.4.2.2.3 Architektur Prototyp

Termin: 1. April 2012

Ein Prototyp der Architektur mit der Schnittstelle zur Datenbank ist fertiggestellt. Erste Version des SAD erstellt.

1.4.2.2.4 End of Elaboration

Termin: 22. April 2012

Die Elaboration Phase ist abgeschlossen. Dies beinhaltet die Fertigstellung des SAD und eines ersten GUI Prototyps.

1.4.2.2.5 Beta Release

Termin: 13. Mai 2012

Ein Beta Release mit allen Grundfunktionen der Web Applikation ist fertiggestellt. Erste Version Benutzerhandbuch ebenfalls erstellt.

1.4.2.2.6 End of Construction

Termin: 3. Juni 2012

Die Phase Construction ist abgeschlossen. Dies beinhaltet die Fertigstellung der Web Applikation mit allen Funktionalitäten. Ausserdem sind das Benutzerhandbuch, die Projektdokumentation und die Erfahrungsberichte grösstenteils bis komplett abgeschlossen.

Erste Versionen der Abgaben sollten schon vor dem 1. Juni 2012 existieren, da ein Feedback von Seiten des Betreuers nur bis am 1. Juni 2012 möglich ist.

1.4.2.2.7 1. Abgabe

Termin: 8. Juni 2012

Beinhaltet die Abgabe der Kurzbeschreibung und des A0-Poster. Erste Versionen der Abgaben für die zweite Abgabe sind erstellt. Feedback von Betreuer nur bis 1. Juni 2012 möglich.

1.4.2.2.8 2. Abgabe

Termin: 15. Juni 2012, 12:00 Uhr

Beinhaltet die Abgabe der Arbeit an den Betreuer bis 12:00 Uhr, die Fertigstellung des A0-Posters bis 12:00 Uhr und deren Abgabe im Abteilungssekretariat (6.113). Ausserdem ist die Schlusspräsentation fertiggestellt. Feedback von Betreuer nur bis 1. Juni 2012 möglich.

1.4.3 Besprechungen

Das Projektteam trifft sich wöchentlich für eine Stunde um mögliche Probleme zu besprechen, das weitere Vorgehen zu planen und Erledigtes zusammenführen.

Ausserdem trifft sich das Projektteam wöchentlich für ebenfalls eine Stunde mit dem Betreuer der Bachelorarbeit um den aktuellen Stand zu besprechen und allfällige Probleme zu klären.

Ausserordentliche Besprechungen werden spontan geplant und abgehalten.

1.4.4 Releases

Nachfolgend werden die geplanten Releases der Software aufgeführt:

- **Architektur Prototyp** – Schnittstelle mit der Datenbank ist aufgebaut.
  (Meilenstein *Architektur Prototyp*)

- **GUI Prototyp** – Das User Interface mit Grundfunktionalitäten ist implementiert.
  (Meilenstein *End of Elaboration)*

- **Beta Release** – Beta Version des Endproduktes mit den primären Funktionalitäten.
  (Meilenstein *Beta Release*)

- **Final Release** – Endgültiges Produkt mit allen Funktionalitäten.
  (Meilenstein *End of Construction*)

1.5 Risikomanagement

1.5.1 Risiken

Siehe Dokument TechnischeRisiken.pdf.

1.5.2 Umgang mit Risiken

Im Folgenden sind die wichtigsten Massnahmen beschreiben die beim Eintreten von Risiken angewendet werden oder Massnahmen um Risiken zu vermeiden.

- Um keinen falschen oder unvorbereiteten Technologie / Container Entscheid zu fällen wird bereits in der Phase Inception und der ersten Iteration der Phase Elaboration ein Studium der möglichen Technologien / Container durchgeführt und ein entsprechender Entscheid gefällt. Dies verringert die Möglichkeit später auftauchender Probleme und mit der Entwicklung kann zum geplanten Zeitpunkt begonnen werden.
- Bei den fehlenden Fachkenntnissen im Bereich JSF ist ein frühzeitiges Einarbeiten in den Problembereich bereits in der Elaboration Phase notwendig. So können verschiedene Techniken oder Implementierungsarten analysiert werden und man kann sich für die sinnvollste für das Projekt entscheiden.
- Ein Ausfall des virtuellen Servers birgt nur ein geringes Risiko und es kann keine direkte Vorbeugung stattfinden. Trotzdem werden alle Daten auf dem SVN Server gesichert und ein wöchentliches Backup der Redmine Daten findet statt, da ansonsten die komplette Projektplanung und Aufwandsverbuchung beim Eintreffen des Risikos verloren gehen würde.
- Um Unit Tests gegen die Datenbank laufen lassen zu können muss mit Mockups gearbeitet werden. Es ist wiederum wichtig frühzeitig sich ins Thema einzuarbeiten um sich nach möglichen Varianten der Implementierung zu erkundigen. Dies geschieht bereits in der Elaboration Phase.

1.6 Arbeitspakete

Die einzelnen Arbeitspakete werden mittels Redmine fortlaufend geplant.

Zu Beginn einer Iteration werden dazugehörige Arbeitspakete beschrieben und auf die einzelnen Projektmitarbeiter aufgeteilt. Die Planung dieser Iterationen und allfällige Erläuterungen dazu werden in den jeweiligen Dokumenten zur Iterationsplanung beschrieben. Zusätzlich findet am Ende einer Iteration ein Assessment statt, welches ebenfalls dokumentiert wird.

Die Redmine Umgebung unseres Projektes befindet sich unter folgender URL:

URL: http://sinv-56024.edu.hsr.ch/redmine

Login: guest / Passwort: guest

1.7 Infrastruktur

Tools zur Entwicklung

- Eclipse
- Ant
- EclEmma (Eclipse Plug-In)
- STAN (Eclipse Plug-In)
- Eclipse Metrics (Eclipse Plug-In)
- FindBugs (Eclipse Plug-In)
- SQL Datenbank
- Subversion (SVN) Server
- Redmine
- Jenkins
- Javadoc

Geräte

- 2 Arbeitsplatzrechner der Schule
- Private Laptops
- SVN Server der Schule
- Virtueller Server mit Jenkins, Redmine und Apache

Dokumentation

- Microsoft Office 2007 / 2010
- Enterprise Architect (UML)
- Balsamiq Mockups (User Interface)

1.8 Qualitätsmassnahmen

Folgende Tabelle gibt einen Überblick über die einzelnen Qualitätsmassnahmen und deren Ziele. Einzelne Massnahmen sind während dem ganzen Projekt relevant, andere nur in bestimmten Phasen.

Massnahme	Zeitraum	Ziel
Fortlaufende Zeiterfassung	ganzes Projekt	ständiger Ist / Soll Abgleich
Reviews im ganzen Team (Statusberichte, Risikoevaluierung, Planung, Protokoll)	min. einmal pro Woche	ständige Überwachung des Projektstatus
Frühzeitiges Durchführen von Technologie- / Containerstudien	Inception / Elaboration I	Auswertung der Technologien / Container und Entscheidung für eine Variante zu Beginn des Projektes (weniger Aufwand / Probleme später)
Wöchentliches Backup von Redmine Daten	einmal pro Woche	Bei einem Ausfall des Server ist der maximale Verlust an Daten auf eine Woche beschränkt
Code Style Guidelines (Naming, Format, usw.)	Elaboration (kurz vor Entwicklungsbeginn)	Einheitliche Terminologie, hohe Lesbarkeit des Codes
Detaillierte Analyse und Design	Elaboration I	Sehr gute Kenntnisse der Problemdomain als Grundstein für solides Software Design legen; Minimierung der Risiken
Review Code	Elaboration / Construction	Review des Codes um Haltung an Style Guidelines und Kommentierung zu gewährleisten.
Bewährte PM und SE Techniken anwenden (z.B. RUP, UML und Patterns)	ganzes Projekt	Aus eigenen Erfahrungen und Erfahrungen anderer lernen und damit effizienter arbeiten; Qualitätsniveau erhöhen

Tabelle 48: Projektplan: Qualitätsmassnahmen

1.8.1 Dokumentation

Alle Dokumente sind für alle Teammitglieder erreichbar und auf einem SVN Server abgelegt. Dadurch können die Dokumente fortlaufend Quergelesen werden. Für jede Abgabe wird jemand verantwortlich gesetzt um die Qualität der Dokumente sicherzustellen und alles für die nötige Abgabe vorzubereiten.

1.8.2 Projektmanagement

Für die Planung und das Management des Projektes und deren einzelnen Iterationen wird Redmine eingesetzt. Redmine dient zur Erfassung der Meilensteine und zur Planung der einzelnen Iterationen. Zu Beginn jeder Iteration werden die dazugehörigen Arbeitspakete eingeplant und auf die Teammitglieder des Projektes aufgeteilt. Ausserdem wird Redmine zur Zeiterfassung verwendet.

Die Redmine Umgebung unseres Projektes befindet sich unter folgender URL:

URL: http://sinv-56024.edu.hsr.ch/redmine

Login: guest / Passwort: guest

1.8.3 Entwicklung

Der Source Code des Projektes wird ebenfalls auf einem SVN Server abgelegt, parallel zu der Dokumentation. Die Qualität des Codes wird durch Redmine sichergestellt mittels der Erfassung von Bugs und Tickets, welche jeweils für die momentane Entwicklungsphase relevant sind.

1.8.3.1 Vorgehen

Die einzelnen Komponenten werden Bottom-Up implementiert. Jede einzelne Komponente ist mit Unit Tests testbar (siehe nächstes Kapitel).

Begonnen wird mit der Datenzugriffsschicht (Data Access Layer, DAL), dies beinhaltet die Schnittstelle zur Datenbank und benötigte Abfragen. Danach kommt die Geschäftslogikschicht (Business Logic Layer, BLL), welche als Schnittstelle für das User Interface die Use Cases auf den DAL abbildet. Die letzte Schicht, das Grafische User Interface (GUI), greift auf den BLL zu und ist daher nur bedingt mit Unit Tests testbar. Auf dieser Ebene werden manuelle Benutzbarkeits- und Integrationstests durchgeführt.

1.8.3.2 Kontinuierliche Integration

Mittels eines Jenkins Build Servers wird die kontinuierliche Integration der Applikation sichergestellt. Bei jedem zentralen Build mittels Jenkins werden automatisch alle Unit Tests ausgeführt, die Testabdeckung wird berechnet und es können auch noch weitere Analysen durchgeführt werden.

1.8.3.3 Unit Testing

Bei der Entwicklung jeder Schicht wird der Ansatz des Test Driven Development verfolgt.

Vor jedem SVN Commit muss lokal eine funktionsfähige Version vorliegen (alle Unit Tests auf grün); dies ist vor allem wichtig, wenn mehr als eine Person im selben Arbeitspaket involviert ist.

Die Testabdeckung der einzelnen Pakte wird mit EclEmma sichergestellt. EclEmma ist ein kostenfreies Tool zur Messung der Testabdeckung in Java Programmen und kann als Plug-In direkt in der Entwicklungsumgebung (z.B. Eclipse) verwendet werden.

1.8.3.4 Code Reviews

Alle Teammitglieder sind gefordert während der Entwicklung auch die Bereiche des anderen Teammitglieds zu überfliegen und falls nötig zu verbessern. Bei grösseren Anpassungen wird der angepasste Code vom jeweils anderen Teammitglied nochmals überprüft um eine hohe Qualität sicherzustellen.

1.8.3.5 Code Style Guidelines

Es werden die gewohnten und in der Java Community etablierten Konventionen verwendet.

Zur Dokumentation des Source Codes wird Javadoc verwendet, es müssen daher Javadoc Comments verwendet werden, damit die Dokumentation automatisch generiert werden kann. Alle Kommentare werden in Englisch verfasst.

Die Javadoc Kommentare sollten folgendes umfassen:

- Name und Beschreibung der Klasse
- Beschreibung von wichtigen Attributen
- Beschreibung aller Methoden, Parametern und Rückgabewerten (kurze oder umfassende Beschreibung, wo es nötig ist)

1.8.4 Testen

1.8.4.1 Systemtest

Das komplette System und all deren Funktionalitäten werden vollumfänglich getestet. Die einzelnen Systemtests werden sauber protokolliert und dokumentiert. Der Ablauf hängt wiederum vom Stand der Implementierung ab und wird jeweils vor der Ausführung aufgestellt.

Diese Systemtests werden laufend durchgeführt (mindestens nach jedem Release). Abschliessend wird nach Fertigstellung der Software ein kompletter Systemtest durchgeführt um die Qualität des Final Releases zu gewährleisten.

2. Technische Risiken

Nr	Titel	Beschreibung	max. Schaden [h]	Eintrittswahrscheinlichkeit	Gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R1	Entscheid Technologie / Container (z.B. Glassfish, Jboss)	Entscheid für entsprechende Technologie / Container sollte möglichst früh gefällt werden um spätere Probleme zu verhindern	40	20%	8	Technologien- / Containerstudium in Inception und Elaboration I	Rasche Lösungsfindung fokussieren, Betreuer fragen
R2	Fehlende Fachkenntnisse (JSF)	Aufgrund fehlender Fachkenntnisse im Bereich JSF dauert die Arbeit länger	25	40%	10	Saubere Analyse und Einarbeitung in den Bereich JSF in Elaboration Phase	Gegenseitige Hilfestellung, Crash-Kurse, Betreuer fragen
R3	Integration Web Applikation	Probleme bei der Integration der Web Applikation auf dem Server	45	30%	13.5	Erste Versuche bereits in Elaboration Phase	Alternative Lösungen suchen, Betreuer fragen
R4	Ausfall Server	Der virtuelle Server fällt aus	90	10%	9	Daten auf SVN sichern und wöchentliches Backup von Redmine (Erneutes Setup kann nicht umgangen werden)	Anderen Server anfordern und neu aufsetzen oder Probleme mit Anbieter direkt klären
R5	Unit Tests für Datenzugriffe	Die Unit Tests für die Zugriffe auf die Datenbank sollten unabhängig von Server laufen	20	40%	8	Suche nach möglichen Technologien bereits in Elaboration Phase	Betreuer um Rat fragen oder lokale Datenbank einrichten
Summe			**220**		**48.5**		

3. Iterationspläne und –assessments

Auf die Pläne und Assessments der einzelnen Iterationen wird in Anhang B verwiesen.

4. Zeitauswertung

In den folgenden Abschnitten werden einige Zeitauswertungen mittels der Daten von Redmine dargestellt und erläutert. Weitere Informationen zu den Aufwänden können direkt unter Redmine nachgeschlagen werden.

Die Redmine Umgebung unseres Projektes befindet sich unter folgender URL:

URL: http://sinv-56024.edu.hsr.ch/redmine

Login: guest / Passwort: guest

4.1 Disziplinen

Die folgende Abbildung zeigt die Aufwände (in Stunden) verteilt auf die einzelnen Disziplinen (Ticket Kategorien) und dies über das gesamte Projekt hinweg.

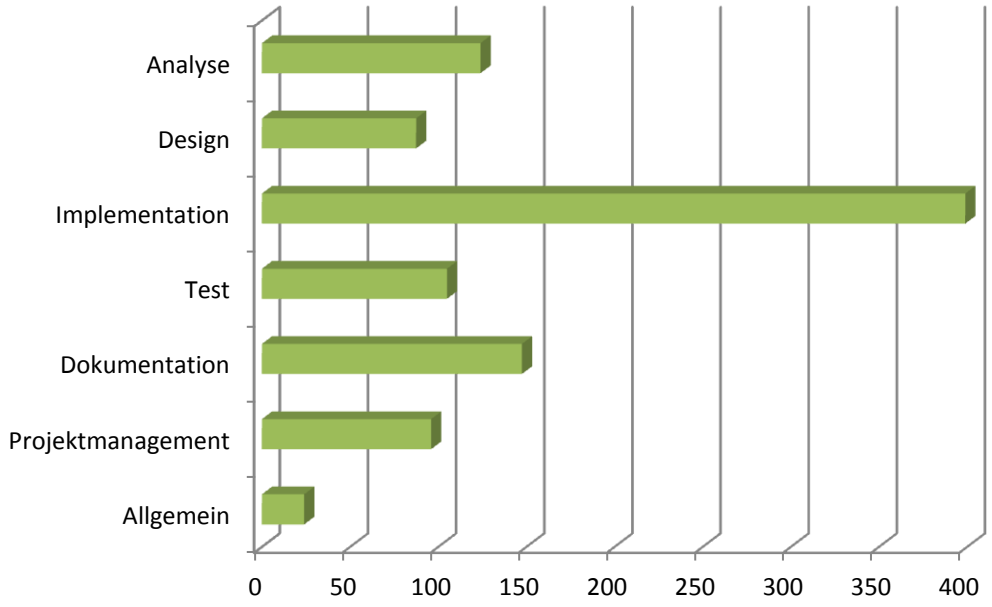

Abbildung 47: Zeitauswertung: Auswertung Disziplinen

Ein Grossteil der Aufwände wurde in die Implementation der Software hineingesteckt. Diese beinhaltet jedoch auch die meisten Entwicklungen von Unit Tests, welche dann in der Disziplin Test fehlen. Von der Komplexität der Software war jedoch anzunehmen, dass die Implementierung die meiste Zeit verbrauchen wird.

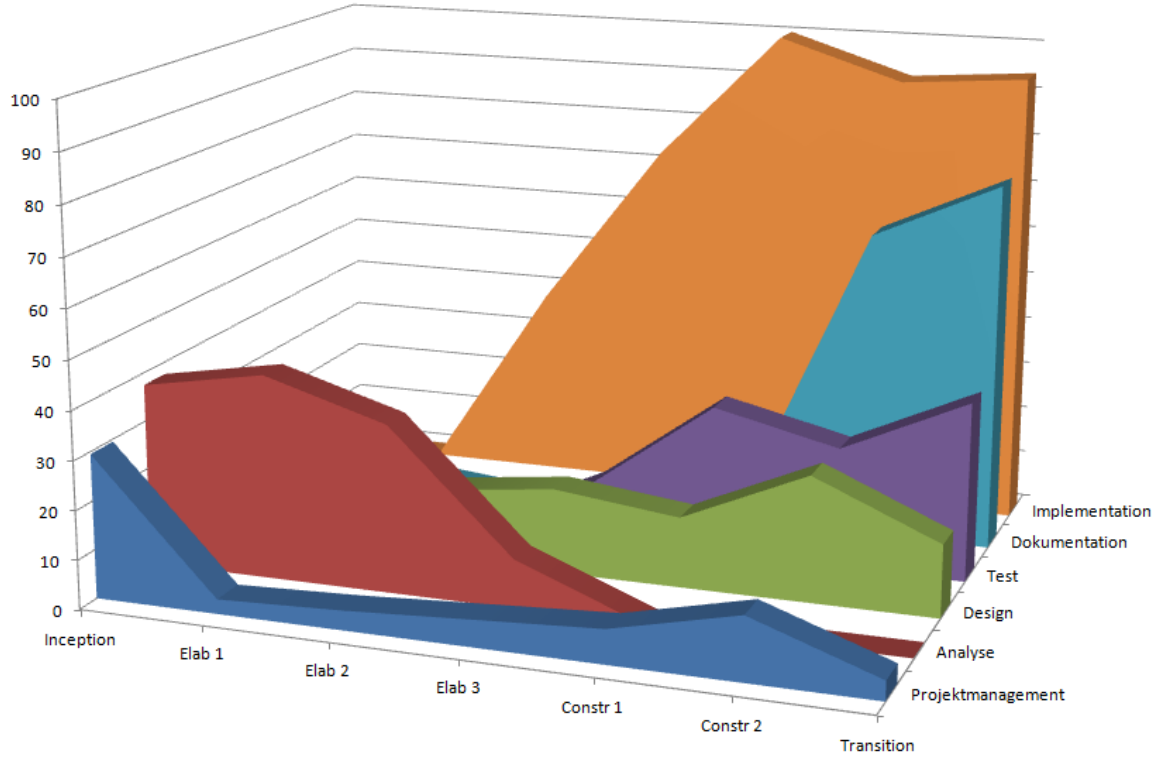

Abbildung 48: Zeitauswertung: Auswertung Iterationen/Disziplinen

In der obigen Grafik sind die einzelnen Disziplinen und deren gebuchten Aufwände verteilt auf die Iterationen während des Projektes erkennbar.

Sie zeigt eine typische Aufteilung für Projekte, welche nach RUP durchgeführt werden. Auch in dieser Abbildung wird gut ersichtlich, wie viel Zeit in die Entwicklung (Implementation) der Software investiert wurde.

4.2 Mitglieder

Die Aufwände der Mitglieder innerhalb der Iterationen sind in folgendem Diagramm ersichtlich.

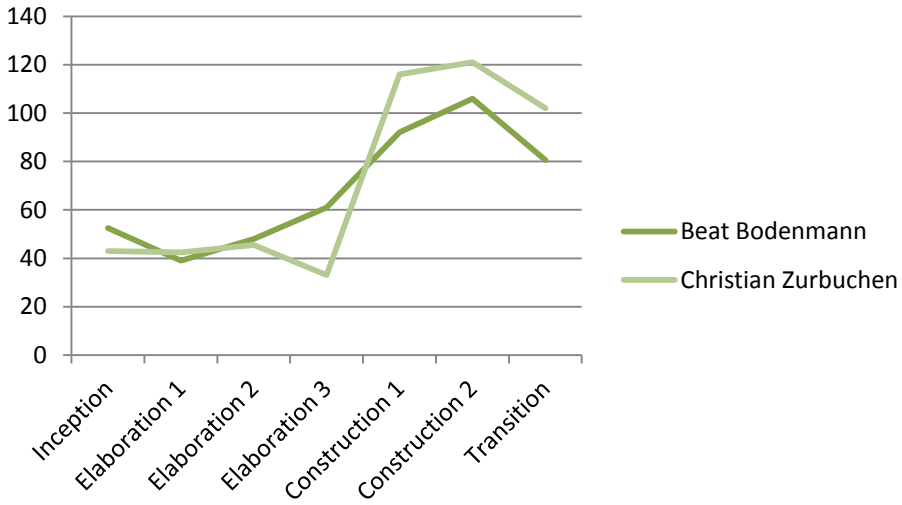

Abbildung 49: Zeitauswertung: Auswertung Mitglieder

Durch den Unterschied in den Aufwänden erkennt man, dass Christian Zurbuchen seine Ferienwoche in der dritten Iteration der Elaboration Phase und Beat Bodenmann diese Woche verteilt auf die letzten drei Iterationen (Construction 1 und 2, Transition) eingezogen hat.

Qualitätssicherung

1. Test

1.1 Einführung

Im Abschnitt *Systemtestspezifikation* befindet sich die Spezifizierung und genaue Beschreibung aller Systemtests. Es handelt sich dabei um die Spezifikation dieser Tests. Den Verweis auf die Protokolle zur Durchführung dieser Tests befindet sich unter Anhang B.

Die Testabdeckung des Source Codes mittels Unit Tests wird im Abschnitt *Testabdeckung* beschrieben.

1.2 Systemtestspezifikation

1.2.1 Voraussetzungen

Hierbei handelt es sich um einen abschliessenden Systemtest, es wird eine funktionsfähige Umgebung für die Durchführung der Tests vorausgesetzt. Diese beinhaltet folgendes:

- Laufender GlassFish Server inklusive Java Datenbank
- PC mit Webbrowser mit Zugriff auf den GlassFish Server

1.2.2 Systemtest

1.2.2.1 Login & Logout

1.2.2.1.1 Vorbereitung

- Die Datenbank muss für die folgenden Tests leer sein.

1.2.2.1.2 Test

Nr.	Beschreibung	Erwartetes Resultat
1	Start der Applikation	User Interface der Applikation erscheint; gesperrte Navigation und Anmeldeseite
2	Einloggen mit ungültigem Benutzer	Fehlermeldung wird angezeigt; Navigation bleibt gesperrt
3	Einloggen mit ungültigem Passwort	Fehlermeldung wird angezeigt; Navigation bleibt gesperrt
4	Einloggen mit dem Default Administrator (admin / admin)	Startseite wird angezeigt und die Navigation ist nicht gesperrt; ausgenommen der Punkt Meine Projekte
5	Logout durchführen	Anmeldeseite erscheint und Navigation ist gesperrt

Tabelle 49: Systemtestspezifikation: Test (Login & Logout)

1.2.2.2 Administration

1.2.2.2.1 Vorbereitung

- Login mit Default Administrator (admin / admin).
- Auf der Datenbank dürfen keine Absenztypen eingetragen sein.

1.2.2.2.2 Test

1.2.2.2.2.1 Passwort ändern

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Passwort ändern* auswählen	Der *Passwort ändern* Bildschirm wird angezeigt
2	Ändern ohne Angaben	Fehlermeldung wird angezeigt
3	Ändern mit falschem alten Passwort; neues Passwort mit mehr als 4 Zeichen	Fehlermeldung wird angezeigt
4	Ändern mit neuem Passwort mit weniger als 4 Zeichen	Fehlermeldung wird angezeigt
5	Ändern ohne neues Passwort	Fehlermeldung wird angezeigt
6	Ändern mit nicht übereinstimmendem Wiederholungspasswort	Fehlermeldung wird angezeigt
7	Ändern mit korrekten Angaben	Erfolgsmeldung wird angezeigt
8	Logout wählen	Loginmaske erscheint und Navigation ist gesperrt
9	Einloggen mit altem Passwort	Fehlermeldung wird angezeigt; Navigation bleibt gesperrt
10	Einloggen mit neuem Passwort	Startseite wird Angezeigt und die Navigation ist nicht gesperrt; ausgenommen der Punkt Meine Projekte

Tabelle 50: Systemtestspezifikation: Test (Passwort ändern)

1.2.2.2.2.2 Absenztypen verwalten

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Absenztypen verwalten* auswählen	Der *Absenztypen verwalten* Bildschirm wird angezeigt
2	Plus wählen	Neue Leere Zeile erscheint mit weisser Farbe
3	Namen (test) eintragen	
4	Farbe wählen	Farbauswahl erscheint und es kann eine Farbe ausgewählt werden
5	Speichern wählen	Erfolgsmeldung wird angezeigt
6	Plus wählen, Name(test) eintragen und Speichern wählen	Fehlermeldung erscheint
7	Farbe des falschen Typen anpassen und Name ändern (test2)	Erfolgsmeldung erscheint und Farbe nur bei falschem Typen anpassen, nicht bei dem Originalen
8	Zwei neue beliebige Typen hinzufügen (test3 und test4) und Speichern wählen	Erfolgsmeldung erscheint

Tabelle 51: Systemtestspezifikation: Test (Absenztypen verwalten)

1.2.2.3 Mitarbeiter

1.2.2.3.1 Vorbereitung

- Login mit Default Administrator (admin / admin).
- Vorab einen neuen Mitarbeiter mit Benutzername *test* erstellen.

1.2.2.3.2 Test

1.2.2.3.2.1 Neuer Mitarbeiter

Anmerkung: In den Test Nr. 2 - 11 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Neu* unter *Mitarbeiter* auswählen	Der entsprechende Bildschirm wird angezeigt
2	Eines der Pflichtfelder (Benutzername, Passwort, Name, Vorname) ist nicht ausgefüllt	Fehlermeldung wird angezeigt
3	Benutzername mit Sonderzeichen	Fehlermeldung wird angezeigt
4	Passwort mit Sonderzeichen	Fehlermeldung wird angezeigt
5	Name, Vorname, Strasse, Ort, Land mit falschem Format (nur Buchstaben, Leerzeichen, ., (,), ‘ oder -)	Fehlermeldung wird angezeigt
6	Nummer der Strasse mit falschem Format (nur Buchstaben, Zahlen, Leerzeichen, . oder -)	Fehlermeldung wird angezeigt
7	PLZ mit falschem Format (min. 3 bis max. 6 Zahlen)	Fehlermeldung wird angezeigt
8	Telefonnummer privat oder mobil mit Buchstaben oder Sonderzeichen (z.B. &,%)	Fehlermeldung wird angezeigt
9	E-Mail-Adresse mit fehlerhaftem Format	Fehlermeldung wird angezeigt
10	Alle Pflichtfelder korrekt ausfüllen mit dem Benutzernamen *test*	Fehlermeldung wird angezeigt
11	Benutzername auf *test2* ändern	Erfolgsmeldung wird angezeigt

Tabelle 52: Systemtestspezifikation: Test (Neuer Mitarbeiter)

1.2.2.3.2.2 Meine Daten

Anmerkung: In den Test Nr. 2 - 11 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Meine Daten* unter *Mitarbeiter* auswählen	Der entsprechende Bildschirm wird angezeigt
2	Eines der Pflichtfelder (Benutzername, Name, Vorname) ist nicht ausgefüllt	Fehlermeldung wird angezeigt
3	Benutzername mit Sonderzeichen	Fehlermeldung wird angezeigt
4	Name, Vorname, Strasse, Ort, Land mit falschem Format (nur Buchstaben, Leerzeichen, ., (,), ‘ oder -)	Fehlermeldung wird angezeigt
5	Nummer der Strasse mit falschem Format (nur Buchstaben, Zahlen, Leerzeichen, . oder -)	Fehlermeldung wird angezeigt
6	PLZ mit falschem Format (min. 3 bis max. 6 Zahlen)	Fehlermeldung wird angezeigt
7	Telefonnummer privat oder mobil mit Buchstaben oder Sonderzeichen (z.B. &,%)	Fehlermeldung wird angezeigt
8	E-Mail-Adresse mit fehlerhaftem Format	Fehlermeldung wird angezeigt
10	Alle Pflichtfelder korrekt ausfüllen und Benutzernamen auf *test* ändern	Fehlermeldung wird angezeigt
11	Benutzername wieder auf *admin* ändern	Erfolgsmeldung wird angezeigt

Tabelle 53: Systemtestspezifikation: Test (Meine Daten)

1.2.2.3.2.3 Mitarbeiter suchen

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Suchen* unter *Mitarbeiter* auswählen	Der entsprechende Bildschirm wird angezeigt
2	Sind weniger als 3 Mitarbeiter auf geführt, weitere Mitarbeiter erstellen	Mitarbeiter werden erstellt und angezeigt im *Suchen* Bildschirm
3	Suche nach Name, Abteilung, Wohnadresse, Wohnort	Suche in allen Spalten werden durchgeführt und Bildschirm wird dynamisch aktualisiert
4	Mitarbeiter auswählen und *Ansicht* wählen	Popup mit Details wird angezeigt; Details gemäss der Erstellungsdaten überprüfen
5	Mitarbeiter auswählen und *Bearbeiten* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird nicht verlassen
6	Mitarbeiter auswählen und *Bearbeiten* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird verlassen und entsprechender Bildschirm wird angezeigt mit Namen des Mitarbeiters im Titel
7	Mitarbeiter auswählen und *Löschen* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Mitarbeiter wird nicht gelöscht
8	Mitarbeiter auswählen und *Löschen* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Mitarbeiter wird gelöscht und erscheint in Liste nicht mehr; Erfolgsmeldung wird ebenfalls angezeigt

Tabelle 54: Systemtestspezifikation: Test (Mitarbeiter suchen)

1.2.2.4 Abteilung

1.2.2.4.1 Vorbereitung

- Login mit Default Administrator (admin / admin).
- Normaler Mitarbeiter mit Benutzername *test* erstellt.

1.2.2.4.2 Test

1.2.2.4.2.1 Neue Abteilung

Anmerkung: In den Test Nr. 2 - 7 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Neu* unter *Abteilung* auswählen	Der entsprechende Bildschirm wird angezeigt
2	Abteilung ohne Namen	Fehlermeldung wird angezeigt
3	Abteilung mit Name *Test*	Erfolgsmeldung wird angezeigt
4	Abteilung mit Name *Test2* und Abteilungsleiter *test*	Als möglicher Abteilungsleiter sollte Mitarbeiter (test) angezeigt werden; Erfolgsmeldung wird angezeigt
5	Abteilung mit Name *Test2.1*, Abteilungsleiter *test* und Oberabteilung *Test2*	Als möglicher Abteilungsleiter sollte Mitarbeiter (test) angezeigt werden; Als mögliche Abteilungen (Unterhalb von) sollten Abteilungen(Test, Test2) angezeigt werden; Erfolgsmeldung wird angezeigt
6	Abteilung mit Name *Test*	Fehlermeldung wird angezeigt
7	Abteilung mit Name *Test2.1.1* unterhalb von Abteilung *Test2.1*; Menüpunkt *Neu* unter *Abteilung* auswählen	Erfolgsmeldung wird angezeigt; Auswahlliste der möglichen Abteilungen (Unterhalb von) ist folgendermassen aufgebaut: Test, Test2, .Test2.1, ..Test2.1.1 um die Level sichtbar zu machen

Tabelle 55: Systemtestspezifikation: Test (Neue Abteilung)

1.2.2.4.2.2 Abteilung suchen

Anmerkungen: Test für neue Abteilungen sollte durchgeführt sein (Abschnitt *Neue Abteilung*). In den Test Nr. 5 & 6 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Suchen* unter *Abteilung* auswählen	Der entsprechende Bildschirm wird angezeigt
2	Abteilung *Test2.1* auswählen und *Ansicht* auswählen	Popup mit Details wird angezeigt; Details gemäss der Erstellungsdaten überprüfen (Anzahl Mitarbeiter = 0)
3	Abteilung auswählen und *Bearbeiten* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird nicht verlassen
4	Abteilung auswählen und *Bearbeiten* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird verlassen und der entsprechende Bildschirm wird angezeigt mit Namen der Abteilung im Titel
5	Name löschen	Fehlermeldung wird angezeigt
6	Namen, Abteilungsleiter und Abteilung (Unterhalb von) ändern	Erfolgsmeldung wird angezeigt
7	2 Mitarbeiter der Abteilung *Test2.1* hinzufügen; Abteilung *Test2.1* auswählen und *Ansicht* auswählen	Anzahl Mitarbeiter hat sich auf 2 erhöht
8	Abteilung *Test* auswählen und *Löschen* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Mitarbeiter wird nicht gelöscht
9	Abteilung *Test* auswählen und *Löschen* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Abteilung wird gelöscht und erscheint in Liste nicht mehr; Erfolgsmeldung wird ebenfalls angezeigt
10	Abteilung *Test2.1* auswählen und *Löschen* wählen; Warnung *Ja* wählen	Abteilung wird gelöscht inklusive Unterabteilungen

Tabelle 56: Systemtestspezifikation: Test (Abteilung suchen)

1.2.2.5 Projekt

1.2.2.5.1 Vorbereitung

- Login mit Default Administrator (admin / admin).
- Normaler Mitarbeiter mit Benutzername *test* erstellt.

1.2.2.5.2 Test

1.2.2.5.2.1 Neues Projekt

Anmerkung: In den Test Nr. 2 - 6 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Neu* unter *Projekte* auswählen	Der entsprechende Bildschirm wird angezeigt
2	Projekt ohne Namen	Fehlermeldung wird angezeigt
3	Projekt ohne Beginn oder Ende	Fehlermeldung wird angezeigt
4	Alle Pflichtfelder korrekt ausfüllen mit Namen *Test*	Erfolgsmeldung wird angezeigt
5	Erneut Projekt mit Namen *Test*	Fehlermeldung wird angezeigt
6	Neues Projekt mit Namen *Test2* und Projektleiter *test*	Erfolgsmeldung wird angezeigt

Tabelle 57: Systemtestspezifikation: Test (Neues Projekt)

1.2.2.5.2.2 Meine Projekte

Anmerkungen: Test für neue Projekte sollte durchgeführt sein (Abschnitt *Neues Projekt*). In den Test Nr. 5 - 7 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Meine Projekte* unter *Projekte* auswählen	Der entsprechende Bildschirm wird angezeigt; Projekte *Test* und *Test2* werden korrekt angezeigt
2	Ein Projekt auswählen und *Ansicht* wählen	Popup mit Details wird angezeigt; Details gemäss der Erstellungsdaten überprüfen (Anzahl Mitarbeiter = 0)
3	Projekt auswählen und *Bearbeiten* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird nicht verlassen
4	Projekt auswählen und *Bearbeiten* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird verlassen und entsprechender Bildschirm wird angezeigt mit Namen des Projektes im Titel
5	Name löschen	Fehlermeldung wird angezeigt
6	Beginn oder Ende löschen	Fehlermeldung wird angezeigt
7	Namen, Beginn und Ende ändern	Erfolgsmeldung wird angezeigt
8	Auf *Projektmitarbeiter* klicken	Seite wird verlassen und entsprechender Bildschirm wird angezeigt mit Namen des Projektes im Titel
9	2 Mitarbeiter auswählen	Mitarbeiter wird nach dem Auswählen nicht mehr in der Suche gefunden, dafür in der Liste angezeigt
10	Beginn von Projektmitarbeiter vor Beginn des Projektes wählen	Fehlermeldung wird angezeigt
11	Ende von Projektmitarbeiter nach Ende des Projektes wählen	Fehlermeldung wird angezeigt
12	Speichern der 2 ausgewählten Mitarbeitern mit korrektem Beginn und Ende	Erfolgsmeldung wird angezeigt
13	*Zurück* Button wählen	Der Bildschirm zum bearbeiten des Projektes wird angezeigt
14	Beim gleichen Projekt erneut *Ansicht* wählen	Anzahl Mitarbeiter hat sich auf 2 erhöht
15	Gleiches Projekt erneut bearbeiten und einen Projektmitarbeiter entfernen; Erneut *Ansicht* wählen	Erfolgsmeldung wird angezeigt; Anzahl Mitarbeiter hat sich auf 1 reduziert
16	Anderes Projekt auswählen und *Löschen* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Projekt wird nicht gelöscht
17	Anderes Projekt auswählen und *Löschen* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Projekt wird gelöscht und erscheint in Liste nicht mehr; Erfolgsmeldung wird ebenfalls angezeigt

Tabelle 58: Systemtestspezifikation: Test (Meine Projekte)

1.2.2.5.2.3 Projekt suchen

Anmerkung: Test für neue Projekte sollte durchgeführt sein (Abschnitt *Neues Projekt*).

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Suchen* unter *Projekte* auswählen	Der entsprechende Bildschirm wird angezeigt; Projekte *Test* und *Test2* werden korrekt angezeigt
2	Suche nach Name, Projektleiter	Suche in allen Spalten werden durchgeführt und Bildschirm wird dynamisch aktualisiert
3	Ein Projekt auswählen und *Ansicht* wählen	Popup mit Details wird angezeigt; Details gemäss der Erstellungsdaten überprüfen (Anzahl Mitarbeiter = 0)
4	Projekt auswählen und *Bearbeiten* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird nicht verlassen
5	Projekt auswählen und *Bearbeiten* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird verlassen und entsprechender Bildschirm wird angezeigt mit Namen des Projektes im Titel
6	Anderes Projekt auswählen und *Löschen* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Projekt wird nicht gelöscht
7	Anderes Projekt auswählen und *Löschen* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Projekt wird gelöscht und erscheint in Liste nicht mehr; Erfolgsmeldung wird ebenfalls angezeigt

Tabelle 59: Systemtestspezifikation: Test (Projekt suchen)

1.2.2.6 Absenzen

1.2.2.6.1 Vorbereitung

- Login mit Default Administrator (admin / admin).
- 2 Absenztypen sind erstellt.

1.2.2.6.2 Test

1.2.2.6.2.1 Neue Absenz (einmalig)

Anmerkung: In den Test Nr. 2 - 9 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Neu* unter *Absenzen* auswählen	Der entsprechende Bildschirm wird angezeigt zum erstellen einer einmaligen Absenz
2	Absenz ohne Beginn (Uhrzeit)	Fehlermeldung wird angezeigt
3	Absenz ohne Ende (Uhrzeit)	Fehlermeldung wird angezeigt
4	Absenz (n. ganztägig); gleicher Tag; Beginn (Uhrzeit) nach Ende (Uhrzeit)	Fehlermeldung wird angezeigt
5	Absenz (n. ganztägig); gleicher Tag; Beginn (Uhrzeit) gleich Ende (Uhrzeit)	Fehlermeldung wird angezeigt
6	Absenz (n. ganztägig); Beginn (Tag) nach Ende (Tag); Uhrzeit egal	Fehlermeldung wird angezeigt
7	Absenz (n. ganztägig); gleicher Tag; Beginn (Uhrzeit) nach Ende (Uhrzeit); Eingabe einer Notiz	Erfolgsmeldung wird angezeigt
8	Absenz (ganztägig); Beginn (Tag) nach Ende (Tag)	Fehlermeldung wird angezeigt
9	Absenz (ganztägig); gleicher Tag	Erfolgsmeldung wird angezeigt

Tabelle 60: Systemtestspezifikation: Test (Neue Absenz, einmalig)

1.2.2.6.2.2 Neue Absenz (periodisch)

Anmerkung: In den Test Nr. 3 - 11 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Neu* unter *Absenzen* auswählen	Der entsprechende Bildschirm wird angezeigt zum erstellen einer einmaligen Absenz
2	Den Button *periodisch* wählen	Der entsprechende Bildschirm wird angezeigt zum erstellen einer periodischen Absenz
3	Absenz ohne Beginn (Uhrzeit)	Fehlermeldung wird angezeigt
4	Absenz ohne Ende (Uhrzeit)	Fehlermeldung wird angezeigt
5	Absenz mit Ende der Periode vor Beginn der Absenz; ganztägige Absenz; gleiche Tage	Fehlermeldung wird angezeigt
6	Absenz mit Ende der Periode vor Ende der Absenz; ganztägige Absenz; verschiedene Tage	Fehlermeldung wird angezeigt
7	Absenz mit Dauer grösser als 1 Tag; täglich wiederkehrend	Fehlermeldung wird angezeigt
8	Absenz mit Dauer grösser als 1 Woche; wöchentlich wiederkehrend	Fehlermeldung wird angezeigt
9	Absenz mit Dauer grösser als 1 Monat; monatlich wiederkehrend	Fehlermeldung wird angezeigt
10	Absenz mit Dauer grösser als 1 Jahr; jährlich wiederkehrend	Fehlermeldung wird angezeigt
11	Korrekte Absenz erfassen inklusive Notiz	Erfolgsmeldung wird angezeigt

Tabelle 61: Systemtestspezifikation: Test (Neue Absenz, periodisch)

1.2.2.6.2.3 Meine Absenzen (einmalig)

Anmerkungen: Test für neue einmalige Absenzen sollte durchgeführt sein (Abschnitt *Neue Absenz (einmalig)*).In den Test Nr. 3 - 11 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Meine Absenzen* unter *Absenzen* auswählen	Der entsprechende Bildschirm wird angezeigt; zuvor erstellte einmalige Absenzen sind in der Liste sichtbar
2	Eine Absenz auswählen und *Ansicht* wählen	Popup mit Details wird angezeigt; Details gemäss der Erstellungsdaten überprüfen
3	Absenz auswählen und *Bearbeiten* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird nicht verlassen
4	Absenz auswählen und *Bearbeiten* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird verlassen und entsprechender Bildschirm wird angezeigt zum Bearbeiten der ausgewählten Absenz
5	Absenz ohne Beginn (Uhrzeit)	Fehlermeldung wird angezeigt
6	Absenz ohne Ende (Uhrzeit)	Fehlermeldung wird angezeigt
7	Absenz (n. ganztägig); gleicher Tag; Beginn (Uhrzeit) nach Ende (Uhrzeit)	Fehlermeldung wird angezeigt
8	Absenz (n. ganztägig); gleicher Tag; Beginn (Uhrzeit) gleich Ende (Uhrzeit)	Fehlermeldung wird angezeigt
9	Absenz (n. ganztägig); Beginn (Tag) nach Ende (Tag); Uhrzeit egal	Fehlermeldung wird angezeigt
10	Absenz (ganztägig); Beginn (Tag) nach Ende (Tag)	Fehlermeldung wird angezeigt
11	Absenz korrekt anpassen	Erfolgsmeldung wird angezeigt
12	Absenz auswählen und *Löschen* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Absenz wird nicht gelöscht
13	Absenz auswählen und *Löschen* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Absenz wird gelöscht und erscheint in Liste nicht mehr; Erfolgsmeldung wird ebenfalls angezeigt

Tabelle 62: Systemtestspezifikation: Test (Meine Absenzen, einmalig)

1.2.2.6.2.4 Meine Absenzen (periodisch)

Anmerkungen: Test für neue periodische Absenzen sollte durchgeführt sein (Abschnitt *Neue Absenz (periodisch)*). In den Test Nr. 6 - 14 ist jeweils abschliessend der Button *Speichern* zu drücken.

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Meine Absenzen* unter *Absenzen* auswählen	Der entsprechende Bildschirm wird angezeigt; zuvor erstellte einmalige Absenzen sind in der Liste sichtbar
2	Den Button periodisch wählen	Der entsprechende Bildschirm wird angezeigt; zuvor erstellte periodische Absenzen sind in der Liste sichtbar
3	Eine Absenz auswählen und *Ansicht* wählen	Popup mit Details wird angezeigt; Details gemäss der Erstellungsdaten überprüfen
4	Absenz auswählen und *Bearbeiten* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird nicht verlassen
5	Absenz auswählen und *Bearbeiten* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Seite wird verlassen und entsprechender Bildschirm wird angezeigt zum Bearbeiten der ausgewählten Absenz
6	Absenz ohne Beginn (Uhrzeit)	Fehlermeldung wird angezeigt
7	Absenz ohne Ende (Uhrzeit)	Fehlermeldung wird angezeigt
8	Absenz mit Ende der Periode vor Beginn der Absenz; ganztägige Absenz; gleiche Tage	Fehlermeldung wird angezeigt
9	Absenz mit Ende der Periode vor Ende der Absenz; ganztägige Absenz; verschiedene Tage	Fehlermeldung wird angezeigt
10	Absenz mit Dauer grösser als 1 Tag; täglich wiederkehrend	Fehlermeldung wird angezeigt
11	Absenz mit Dauer grösser als 1 Woche; wöchentlich wiederkehrend	Fehlermeldung wird angezeigt
12	Absenz mit Dauer grösser als 1 Monat; monatlich wiederkehrend	Fehlermeldung wird angezeigt
13	Absenz mit Dauer grösser als 1 Jahr; jährlich wiederkehrend	Fehlermeldung wird angezeigt
14	Absenz korrekt anpassen	Erfolgsmeldung wird angezeigt
15	Absenz auswählen und *Löschen* wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Absenz wird nicht gelöscht
16	Absenz auswählen und *Löschen* wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Absenz wird gelöscht und erscheint in Liste nicht mehr; Erfolgsmeldung wird ebenfalls angezeigt

Tabelle 63: Systemtestspezifikation: Test (Meine Absenzen, periodisch)

1.2.2.6.2.5 Genehmigung

Anmerkungen: Tests für neue Absenzen sollten durchgeführt sein (Abschnitte *Neue Absenz (einmalig)* und *Neue Absenz (periodisch)*).

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Genehmigung* unter *Absenzen* auswählen	Der entsprechende Bildschirm wird angezeigt
2	Den Benutzer mit zuvor erstellen Absenzen aufklappen	Alle zuvor erstellten Absenzen werden dargestellt
3	Das Lupensymbol wählen	Popup mit Details wird angezeigt; Details gemäss der Erstellungsdaten überprüfen
4	Genehmigen (Häkchen-Symbol) wählen; Warnung *Abbrechen* wählen	Popup mit Details zur Genehmigung wird angezeigt; User Interface gesperrt; Seite wird nicht verlassen
5	Genehmigen (Häkchen-Symbol) wählen; Warnung *Ja* wählen	Popup mit Details zur Genehmigung wird angezeigt; User Interface gesperrt; Absenz nicht mehr in Liste sichtbar
6	Ablehnen (Kreuz-Symbol) wählen; *Abbrechen* wählen	Popup mit Details zur Ablehnung wird angezeigt; User Interface gesperrt; Seite wird nicht verlassen
7	Ablehnen (Kreuz-Symbol) wählen; *Ja* wählen	Popup mit Details zur Ablehnung wird angezeigt; User Interface gesperrt; Absenz nicht mehr in Liste sichtbar
8	Menüpunkt *Meine Absenzen* unter *Absenzen* auswählen	Der entsprechende Bildschirm wird angezeigt; zuvor genehmigte oder abgelehnte Absenz wird als solche dargestellt
9	Die abgelehnte Absenz auswählen und Status zurücksetzen wählen; Warnung *Nein* wählen	Warnung wird angezeigt; User Interface gesperrt; Status wird nicht zurückgesetzt
10	Die abgelehnte Absenz auswählen und Status zurücksetzen wählen; Warnung *Ja* wählen	Warnung wird angezeigt; User Interface gesperrt; Status wird auf *Offen* zurückgesetzt
11	Menüpunkt *Genehmigung* unter *Absenzen* auswählen	Der entsprechende Bildschirm wird angezeigt; zuvor zurückgesetzte Absenz ist wieder in der Liste vorhanden

Tabelle 64: Systemtestspezifikation: Test (Genehmigung)

1.2.2.6.2.6 Übersicht

Folgende Absenzen vorgängig erstellen:

1. Einmalige Absenz (n. ganztägig, gleicher Tag, Uhrzeit variierend)
2. Einmalige Absenz (n. ganztägig, verschiedene Tage, Uhrzeit variierend)
3. Einmalige Absenz (ganztägig, gleicher Tag)
4. Einmalige Absenz (ganztägig, verschiede Tage)
5. Periodische Absenz (n. ganztägig, täglich wiederkehrend, innerhalb eines Monates)
6. Periodische Absenz (ganztägig, täglich wiederkehrend, innerhalb eines Monates)
7. Periodische Absenz (n. ganztägig, wöchentlich wiederkehrend, innerhalb einiger Monate)
8. Periodische Absenz (ganztägig, wöchentlich wiederkehrend, innerhalb einiger Monate)
9. Periodische Absenz (n. ganztägig, monatlich wiederkehrend, über mehrere Schaltjahre hinweg, Tag ist der 31. Januar)
10. Periodische Absenz (ganztägig, monatlich wiederkehrend, über mehrere Schaltjahre hinweg, Tag ist der 29. Februar)
11. Periodische Absenz (n. ganztägig, jährlich wiederkehrend, über mehrere Jahre hinweg)
12. Periodische Absenz (ganztägig, jährlich wiederkehrend, über mehrere Schaltjahre hinweg, Tag ist der 29. Februar)

Nr.	Beschreibung	Erwartetes Resultat
1	Menüpunkt *Übersicht* unter *Absenzen* auswählen	Der entsprechende Bildschirm wird angezeigt
2	Korrekte Abteilung des Mitarbeiters, für welchen die obigen Absenzen erfasst wurden auswählen	Korrekte Anzeige der einzelnen Absenzen. Speziell zu beachten gilt es: - Absenz 9 sollte monatlich immer am letzten Tag des Monats sein (31, 30, 29 oder 28) - Absenz 10 sollte monatlich immer am 29 Tag sein (ausser 28 bei nicht Schaltjahr Februar) - Absenz 12 sollte immer 29 oder 28 Februar sein (je nach Schaltjahr, letzter Tag im Februar)

Tabelle 65: Systemtestspezifikation: Test (Übersicht)

1.2.2.7 Rollen & Rechte

1.2.2.7.1 Vorbereitung

Folgende Struktur Aufbauen:

- Abteilung 1, Abteilungsleiter Mitarbeiter 1
  - Mitarbeiter 1
  - Unterabteileung 1.1, Abteilungsleiter Mitarbeiter 1.1
    - Mitarbeiter 1.1.1
    - Mitarbeiter 1.1.2
- Abteilung 2, Abteilungsleiter Mitarbeiter 2
  - Mitarbeiter 2
  - Mitarbeiter 2.1
  - Mitarbeiter 2.2
- Projekt 1, Leiter: Mitarbeiter 1.1.1, Manager: Mitarbeiter 1.1
  - Mitglied Mitarbeiter 1.1.2
- Projekt 2, Leiter Mitarbeiter 2.1, Manager: Mitarbeiter 2
  - Mitglied Mitarbeiter 2.2
- Für alle Mitarbeiter diverse Absenzen eintragen (periodische und einmalige)

Diese Vorbereitung soll für die Durchführung der folgenden Tests dienen.

1.2.2.7.2 Test

1.2.2.7.2.1 Mitarbeiter

Nr.	Beschreibung
1	Jeder Mitarbeiter kann seine eigenen Daten lesen und bearbeiten und sich selbst nicht löschen
2	Ein „normaler“ Mitarbeiter kann sich nicht zum Administrator ernennen, die Abteilung oder sein Pensum wechseln; er kann die Daten anderer lesen, diese jedoch nicht bearbeiten oder löschen
3	Ein Abteilungsleiter kann neue Mitarbeiter nur unter seiner Stufe hinzufügen
4	Ein Abteilungsleiter kann zusätzlich die Daten der Mitarbeiter unter seiner Stufe bearbeiten oder diese löschen
5	Der Administrator kann sämtliche Daten aller Mitarbeiter bearbeiten oder diese löschen

Tabelle 66: Systemtestspezifikation: Test (Rollen & Rechte, Mitarbeiter)

1.2.2.7.2.2 Abteilungen

Nr.	Beschreibung
1	Jeder Mitarbeiter kann die Abteilungen lesen
2	Ein „normaler“ Mitarbeiter kann nur lesen
3	Ein Abteilungsleiter kann neue Abteilungen nur unterhalb seiner Stufe erfassen
4	Ein Abteilungsleiter kann die Daten seiner Abteilung und deren Unterabteilung bearbeiten oder Abteilungen löschen
5	Der Administrator kann sämtliche Daten bearbeiten, Abteilungen löschen und neu hinzufügen (auch Superabteilungen)

Tabelle 67: Systemtestspezifikation: Test (Rollen & Rechte, Abteilung)

1.2.2.7.2.3 Projekte

Nr.	Beschreibung
1	Jeder Mitarbeiter kann die Projekte lesen
2	Ein „normaler“ Mitarbeiter kann nur lesen
3	Ein beliebiger Mitarbeiter kann die Projekte, in welchen er als Manager eingetragen ist, bearbeiten (jedoch nicht den Manager ändern)
4	Ein Projektleiter kann die Daten seiner Projekte bearbeiten (jedoch nicht den Projektleiter oder Manager ändern)
5	Ein Abteilungsleiter kann Projekte neu erstellen, jedoch nur Mitarbeiter unterhalb seiner Stufe als Projektleiter oder Manager einteilen
6	Der Administrator kann sämtliche Daten bearbeiten und auch Projekte entfernen und neu erstellen

Tabelle 68: Systemtestspezifikation: Test (Rollen & Rechte, Projekte)

1.2.2.7.2.4 Absenzen

Nr.	Beschreibung
1	Jeder Mitarbeiter kann seine eigenen Absenzen lesen, entfernen und bearbeiten sofern sie noch nicht genehmigt worden sind; Jeder kann alle Absenzen in der Übersicht lesen
2	Ein Abteilungsleiter kann zusätzlich die Absenzen der Mitarbeiter unter seiner Stufe genehmigen / ablehnen
3	Ein Projektleiter kann die Absenzen der Mitarbeiter in seinem Projekt genehmigen / ablehnen sofern diese nicht periodisch sind und die Dauer eines Tages nicht überschreiten
4	Der Administrator kann sämtliche Absenzen bearbeiten, genehmigen / ablehnen und auch entfernen

Tabelle 69: Systemtestspezifikation: Test (Rollen & Rechte, Absenzen)

1.3 Testabdeckung

Die Testabdeckung der Software wurde mittels Emma für Jenkins ausgewertet. Folgende Abbildung zeigt die Auswertung aller Packages.

Overall Coverage Summary

Name	Klassen	Methoden	Blöcke
all classes	53.4% 31/58	42.4% 450/1062	42.2% 7844/18584

Coverage Breakdown by Package

Name	Klassen	Methoden	Blöcke
bll.bean	0.0% 0/1	0.0% 0/10	0.0% 0/109
bll.bean.absence	0.0% 0/5	0.0% 0/194	0.0% 0/4009
bll.bean.absence.node	0.0% 0/5	0.0% 0/29	0.0% 0/107
bll.bean.admin	0.0% 0/2	0.0% 0/24	0.0% 0/408
bll.bean.controller	66.7% 2/3	93.3% 56/60	94.4% 559/592
bll.bean.controller.util	50.0% 1/2	60.0% 12/20	56.9% 112/197
bll.bean.converter	0.0% 0/1	0.0% 0/3	0.0% 0/28
bll.bean.department	0.0% 0/3	0.0% 0/64	0.0% 0/1058
bll.bean.employee	0.0% 0/3	0.0% 0/101	0.0% 0/1146
bll.bean.project	0.0% 0/5	0.0% 0/91	0.0% 0/1423
bll.domain	100.0% 6/6	62.9% 105/167	43.6% 682/1566
bll.domain.util	100.0% 3/3	87.5% 21/24	96.0% 503/524
bll.exception	100.0% 1/1	100.0% 3/3	100.0% 14/14
bll.handler	100.0% 4/4	100.0% 55/55	64.7% 999/1544
dal.entity	100.0% 7/7	91.6% 109/119	93.6% 875/935
dal.exception	100.0% 1/1	25.0% 1/4	22.2% 4/18
dal.service	100.0% 2/2	95.1% 78/82	83.2% 3914/4703
dal.util	100.0% 4/4	83.3% 10/12	89.7% 182/203

Abbildung 50: Qualitätssicherung: Testabdeckung (alle Packages)

Das Package bll.bean wurde nicht mit Unit Tests abgedeckt, da ein Testen der Managed Beans sehr mühsam und aufwendig ist. Dies erfordert ein Mocking des Faceskontext, was mit JMock möglich wäre. Dieses Mocking wurde aber aus Zeitgründen und auf Grund des sehr ausführlichen Systemtests nicht durchgeführt. Der Systemtest ist dafür verantwortlich, dass alle User Interfaces im Zusammenspiel mit den Beans vollumfänglich funktionieren.

Overall Coverage Summary

Name	Klassen	Methoden	Blöcke
all classes	93.9% 31/33	82.4% 450/546	76.2% 7844/10296

Coverage Breakdown by Package

Name	Klassen	Methoden	Blöcke
bll.bean.controller	66.7% 2/3	93.3% 56/60	94.4% 559/592
bll.bean.controller.util	50.0% 1/2	60.0% 12/20	56.9% 112/197
bll.domain	100.0% 6/6	62.9% 105/167	43.6% 682/1566
bll.domain.util	100.0% 3/3	87.5% 21/24	96.0% 503/524
bll.exception	100.0% 1/1	100.0% 3/3	100.0% 14/14
bll.handler	100.0% 4/4	100.0% 55/55	64.7% 999/1544
dal.entity	100.0% 7/7	91.6% 109/119	93.6% 875/935
dal.exception	100.0% 1/1	25.0% 1/4	22.2% 4/18
dal.service	100.0% 2/2	95.1% 78/82	83.2% 3914/4703
dal.util	100.0% 4/4	83.3% 10/12	89.7% 182/203

Abbildung 51: Qualitätssicherung: Testabdeckung (ohne Package bll.bean)

Die obere Grafik zeigt die Abdeckung ohne das bll.bean Package. Diese Testabdeckung ist hoch und somit akzeptabel für unser Projekt. Bei den Klassen, Methoden und Blöcken, welche nicht abgedeckt sind, handelt es sich entweder um simple Methoden oder um nur sehr umständlich zu mockende Methoden. Für diesen Bereich ist wiederum der Systemtest verantwortlich.

2. Code Analyse

2.1 Kennzahlen

Für die Entwicklung von teamMgmt wurden 72 Klassen implementiert mit einer totalen Anzahl von 3991 Codezeilen. Diese Werte wurden mittels dem Eclipse Metrics Plug-In ermittelt.

2.2 Struktur

Die Software wurde in drei Schichten aufgeteilt:

- **WebContent** - User Interfaces (Ordner)
- **bll** - Geschäftslogik (Package)
- **dal** - Datenzugriffsschicht (Package)

Die Reihenfolge der Zugriffe der einzelnen Ordner / Packages wird in folgender Abbildung ersichtlich.

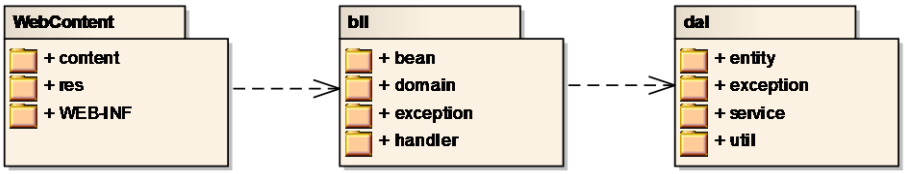

Abbildung 52: Qualitätssicherung: Struktur Software

2.2.1 Package bll

Innerhalb des Package bll sind Zugriffe nur in eine Richtung und es sind keine Zyklen unter den einzelnen Subpackages vorhanden. Die Klassen des bll.bean Packages werden ausschliesslich von den User Interfaces verwende. Von diesen Klassen verläuft der weitere Zugriff innerhalb des Packages. Sie bilden die Schnittstelle zwischen User Interfaces und Geschäftslogik.

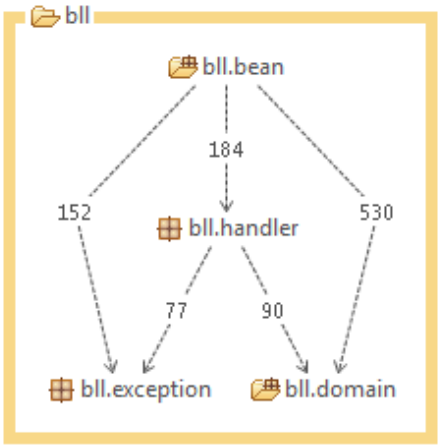

Abbildung 53: Qualitätssicherung: Struktur Package bll

Das Package bll.bean beherbergt alle Managed Beans, welche von den User Interfaces verwendet werden. Im Package bll.bean.controller befinden sich Session Beans, die zur Kontrolle der Navigation und der aktuellen Sitzung dienen. Diese werden von den einzelnen Bean Klassen verwendet.

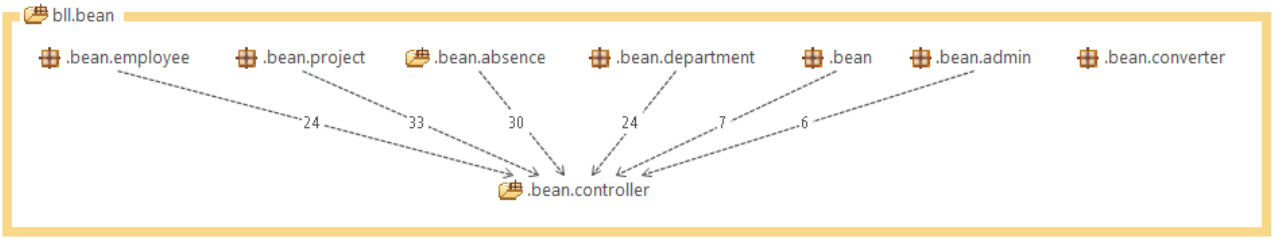

Abbildung 54: Qualitätssicherung: Struktur Package bll.bean

2.2.2 Package dal

Das Package dal besitzt ebenfalls eine klare Struktur mit Zugriffen in lediglich eine Richtung. Die Klassen des dal.service Packages bieten die Schnittstelle für die Geschäftslogik zur Datenbank.

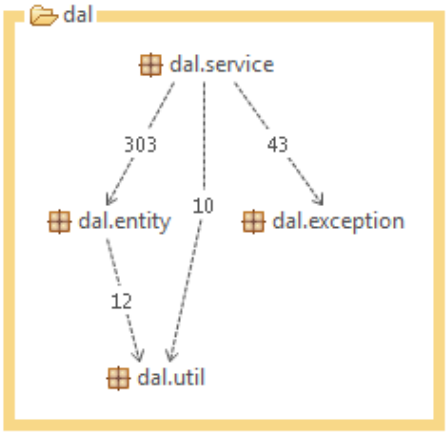

Abbildung 55: Qualitätssicherung: Struktur Package dal

2.3 Metriken

In den folgenden Abschnitten werden einzelne und wichtige Metriken innerhalb der Software analysiert und ausgewertet. Diese Auswahl wurde nach gängigem Gebrauch festgelegt. Die Metriken wurden mit dem Metrics Plug-In für Eclipse ausgewertet. Den Verweis auf die Resultate weiterer Metriken befindet sich unter Anhang B.

2.3.1 Cyclomatic Complexity

Die zyklomatische Komplexität (Cyclomatic Complexity) misst die Komplexität einer Software anhand des Kontrollflusses innerhalb der einzelnen Methoden.

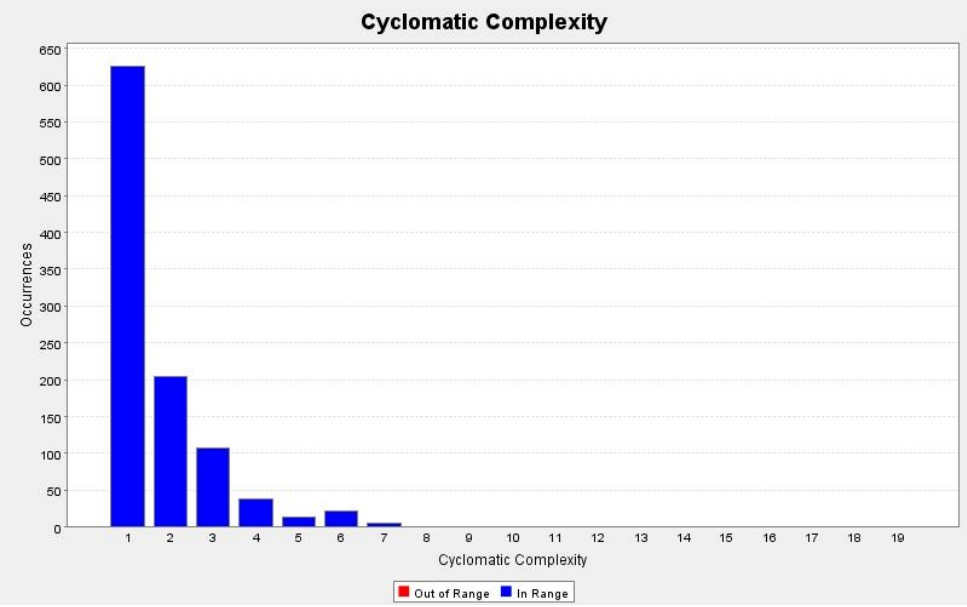

Abbildung 56: Qualitätssicherung: Cyclomatic Complexity

Beinahe alle Klassen befinden sich im Bereich der vertretbaren Komplexität von einem Wert von maximal 10. Sie befinden sich sogar unterhalb einem Wert von 7, ausser einer Klasse, welche den Wert allerdings deutlich überschreitet. Mit dem Wert 19 weist die PersistenceServiceMock Klasse eine sehr hohe Komplexität auf. Dieser sehr hohe Wert taucht jedoch nur in einer Methode auf und diese initialisiert alle Absenzen für den Mock, was für die Lesbarkeit und Wartbarkeit der Methode kein Problem darstellt. Da er nur in einer Methode auftaucht ist er in der Grafik auch nicht ersichtlich.

2.3.2 Feature Envy

Die Metrik Feature Envy überprüft, ob sich eine Methode mehr um für die Daten einer anderen Klasse interessiert als für diejenigen ihrer eigenen Klasse.

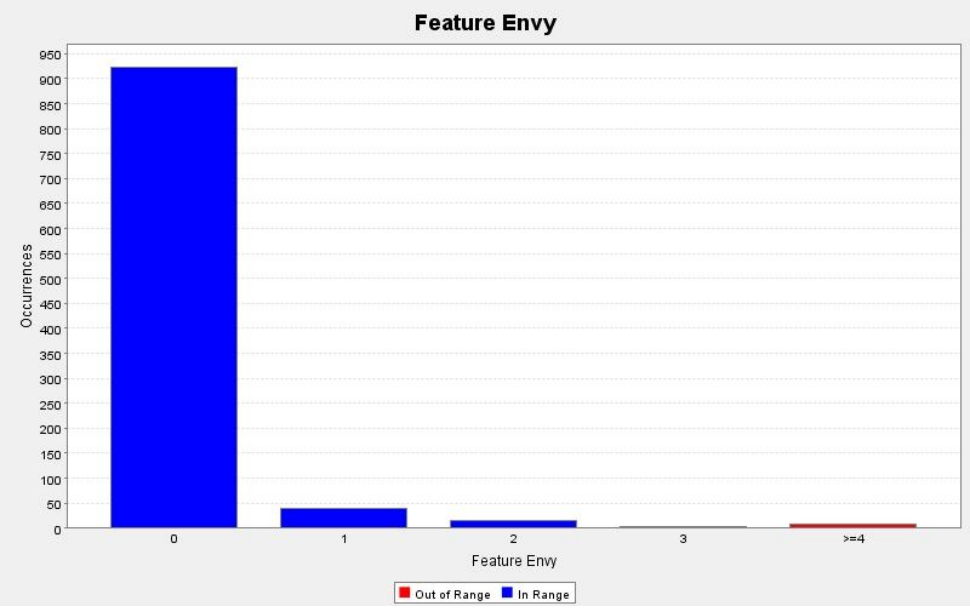

Abbildung 57: Qualitätssicherung: Feature Envy

Wiederum befinden sich fast alle Klassen im grünen Bereich von maximal 2. Die PersistenceServiceMock Klasse beinhaltet Initialisierungsmethoden für die die gemockten Daten, welche auf die Domainklassen zugreifen und somit als negativ auffallen. Ausserdem befinden sich innerhalb des Beans, welches für den Aufbau der Übersicht der Absenzen verantwortlich ist, Methoden die einen leicht zu hohen Wert aufweisen. Es handelt sich dabei um Methoden, die eine periodische Absenz korrekt innerhalb des Kalenders darstellen. Dies wäre nur sehr umständlich innerhalb der Absence Klasse machbar, da die Domainklasse nicht mit Komponenten des User Interfaces zu tun haben sollte. Aus diesen Gründen wurden diese Überschreitungen als unproblematisch eingestuft.

2.3.3 Lines of Code in Method

Mittels dieser Metrik werden die Anzahl Codezeilen (Lines of Code) innerhalb einer Methode ausgewertet. Die Zeilen für Javadoc und Leerzeilen sind nicht Bestandteil der Berechnung. Die Metrik dient zur Bewertung, ob eine Methode einfach lesbar und somit auch wartbarer ist.

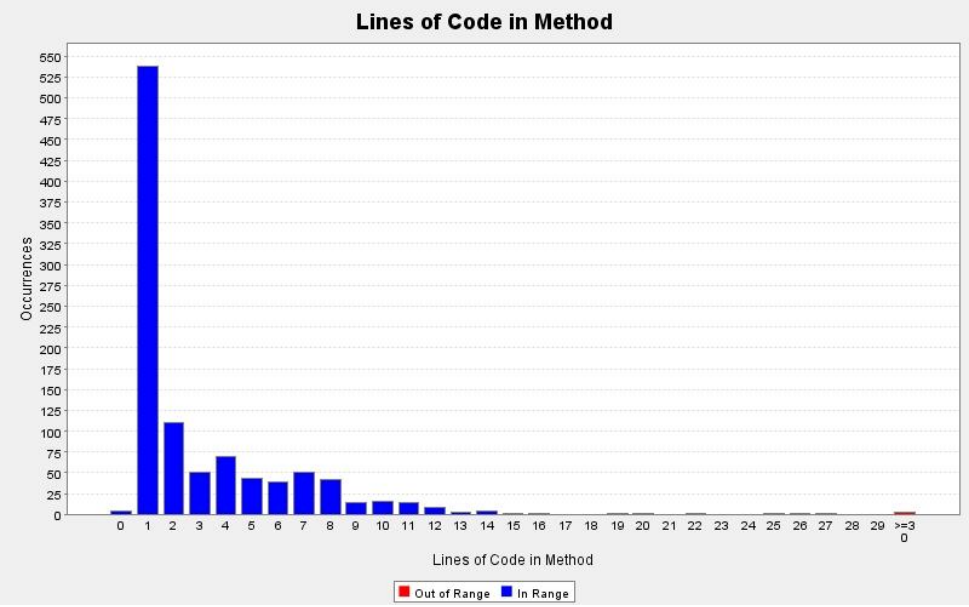

Abbildung 58: Qualitätssicherung: Lines of Code in Method

Der maximale Wert von 15 Codezeilen pro Methode wird in praktisch keiner Klasse überschritten. Wiederum stellt die PersistenceServiceMock Klasse die Ausnahme. Durch viele Initialisierungsmethoden, welche teilweise sehr lang sind, fällt die Klasse negativ auf. Diese Methoden sind jedoch sehr übersichtlich und inhaltlich sehr einfach und können deshalb als unproblematisch angesehen werden.

2.3.4 Efferent Coupling

Bei dieser Metrik handelt es sich um die Berechnung der Anzahl Klassen in einem Package, welche von Klassen ausserhalb des Package abhängen.

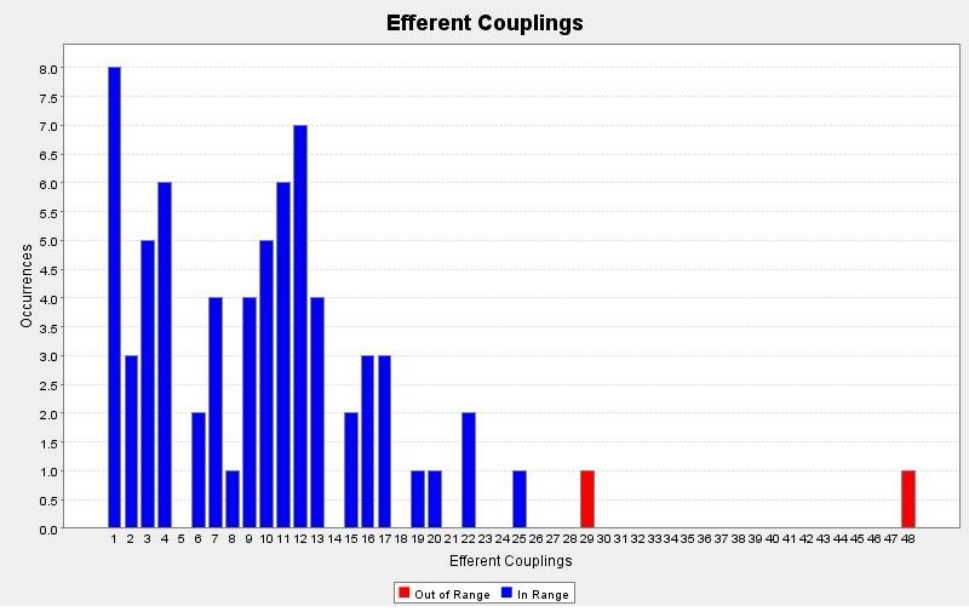

Abbildung 59: Qualitätssicherung: Efferent Coupling

Zwei Klassen fallen bei dieser Metrik negativ auf, und zwar die Klassen PersistenceServiceMock und PersistenceService. Diese dienen als Schnittstelle zur Datenzugriffsschicht und sind als Singleton implementiert. Die Geschäftslogik greift über die Singleton Instanz jeweils auf die gewünschte Methode zu. Somit ist klar, dass diese beiden Schnittstellen bei dieser Metrik einen negativen Wert aufweisen. Sie können jedoch als unproblematisch eingestuft werden.

2.3.5 Lack of Cohesion

Die Metrik für Lack of Cohesion berechnet die Kohäsion innerhalb einer Klasse und deren Methoden.

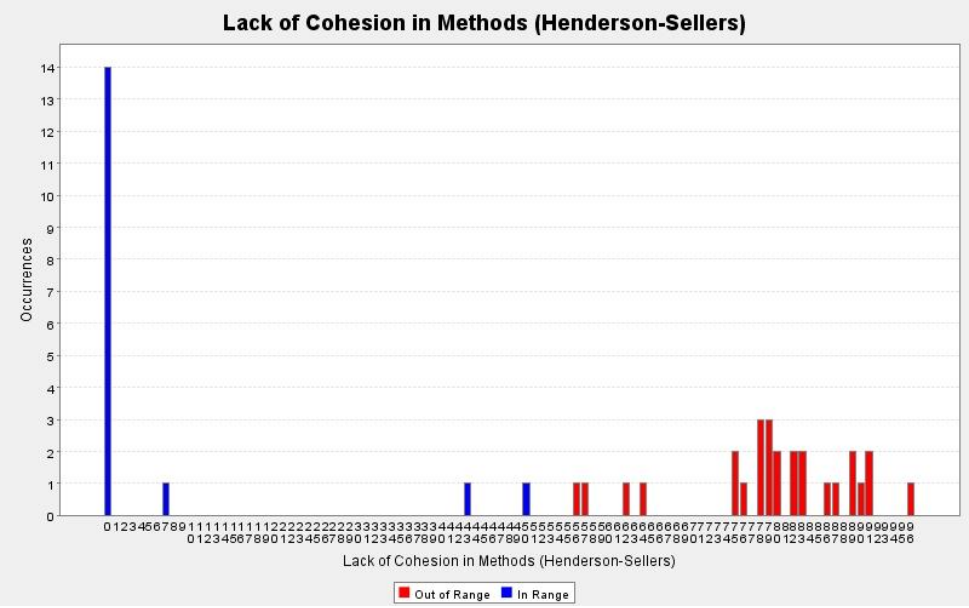

Abbildung 60: Qualitätssicherung: Lack of Cohesion

Das Problem dieser Metrik ist, dass Klassen mit Getter und Setter Methoden die Metrik massiv verschlechtern. Bei den negativ auffallenden Klassen handelt es sich entweder um Managed Bean Klassen oder um Domain / Entity Klassen. Die Managed Bean Klassen verfügen lokal über sehr viele Attribute, welche in den User Interfaces angezeigt werden. Diese Anzeige funktioniert nur über Getter und Setter Methoden. Bei den Domain und Entity Klassen handelt es sich über reine Datenklassen, auf welchen ein direkter Zugriff auf die privaten Attribute nur über Getter und Setter Methoden möglich ist. Diese beiden Arten von Klassen sind ein bekanntes Problem im Zusammenhang mit dieser Metrik und können deshalb ignoriert werden.

2.4 FindBugs

Bei FindBugs handelt es sich um ein frei erhältliches Plug-In für Eclipse. Es sucht innerhalb der Software nach speziellen Fehlermustern, welche meist auf tatsächliche Fehler innerhalb des Programms hinweisen.

Alle Funde von FindBugs nach Fehlermustern in der Software konnten während der Entwicklung und des Refactoring Prozesses ausgemerzt werden.

Erfahrungen Technologien

1. GlassFish

GlassFish ist ein stabiler Applikationsserver der ziemlich einfach zu handhaben ist. Er bietet eine übersichtliche Administrationskonsole, welche via Webbrowser verwendet werden kann. Dadurch wird die Arbeit über das CLI minimiert. Nur zu Beginn, um den Server und die Datenbank zu starten und einzurichten, musste das CLI verwendet werden. Alle weiteren Aktionen konnten über das Webinterface getätigt werden. Beispielsweise kann das Deployment einer Applikation bequem mittels eines Webuploads erledigt werden.

Das Webinterface hat leider ziemlich hohe Ladezeiten, dies könnte jedoch auch von der geringen Leistung der von uns verwendeten Serverumgebung und der ständigen Verwendung eines SSH Tunnels herrühren.

Neben einer einfachen Administration überzeugt GlassFish mit Stabilität und Performance. Abgesehen vom ersten Aufruf nach einem Deployment sind die durchschnittlichen Antwortzeiten mit weniger als einer Sekunde gering. Die Dauer des ersten Aufrufs ist insofern länger, weil die Datenbank und deren Tabellen noch erstellt werden müssen.

Ein weiterer Vorteil von GlassFish ist die integrierte Java Datenbank, welche gerade für eine kleinere Applikation interessant ist. Da sie bereits in den Server eingebunden ist muss man sich weder um Treiber noch um die Verbindung kümmern, sondern man kann die Datenbank einfach starten. In der Applikation muss die Datenbank selbstverständlich konfiguriert werden.

2. EJB & JPA

Für das Projekt teamMgmt wurde nur ein kleiner Teil der Möglichkeiten der Enterprise Java Beans (EJB) ausgeschöpft. Sie werden als Entities verwendet, welche danach mittels der Java Persistence API (JPA) auf der Datenbank persistiert werden. Die Verwendung von JPA bietet insofern einen Vorteil, dass die Benutzung einer Datenbank drastisch vereinfacht wird. Für die Erstellung, Bearbeitung und das Löschen von Objekten müssen keine Abfragen geschrieben werden, da dies JPA übernimmt. Jedes Objekt kann über seiner Identifikation (Primärschlüssel) erreicht und von der Datenbank geladen werden. Einzig für speziellere Abfragen und die Abfrage nach Nicht-Primärschlüsselfeldern müssen Abfragen von Hand geschrieben werden.

Die Verwendung von JPA ist angenehm, nicht nur weil mit Hilfe von Annotationen dies direkt im Code geschieht.

Das persistence.xml ist der zentrale Konfigurationspunkt für JPA. Dort können die Entity Klassen, Datenbanktreiber, Generationsarten und vieles mehr angegeben. Dies wirkte zu Beginn etwas verwirrend, sollte aber keine grösseren Probleme schaffen.

Probleme, die im Zusammenhang mit EJB und JPA auftauchten sind im Abschnitt *Technologische Aspekte* genauer beschrieben.

3. JSF

Die Grundlagen der Entwicklung einer Webapplikation mit der Java Server Faces (JSF) Komponente war aus der Vorlesung Internettechnologien bereits bekannt. Die Verwendung von JSF ist relativ einfach und verfügt über viele brauchbare Nachschlagwerke im Internet.

4. Primefaces

Primefaces ist eine Komponentenframework, welches zur Erweiterung von JSF dient. Es bietet zusätzliche Komponenten für die Entwicklung von User Interfaces an.

Die Verwendung von Primefaces ist sehr einfach, da ein leicht umsetzbarer Showcase online existiert. Dieser Showcase beschreibt die Verwendung aller Komponenten von Primefaces.

Zu Beginn dauerte es etwas länger bis Primefaces innerhalb von JSF verwendet werden kann. Das Problem dabei war die falsche Version von JSF in Kombination mit der zu verwendenden Version von Primefaces. Mit einem einfachen Update der JSF Version war dieses Problem behoben und Primefaces konnte ohne Umstände verwendet werden. Die einzige Schwierigkeit dabei war es herauszufinden, wo das Problem liegt.

Bei der Kombination von Primefaces mit JSF tauchten weitere Probleme bei der Verwendung von gewissen Primefaces Komponenten innerhalb von anderen Primefaces Komponenten oder JSF Komponenten auf. Diese Probleme sind jedoch ausführlich beschrieben im Abschnitt *Technologische Aspekte*.

Ansonsten war die Entwicklung mit Primefaces sehr angenehm und auch online gut dokumentiert. Bei kleineren Problemen konnte rasch eine passable Lösung gefunden und die Arbeit fortgesetzt werden.

Persönliche Berichte

1. Beat Bodenmann

1.1 Projektverlauf

Zu Beginn der Arbeit wendeten wir viel Zeit für die Spezifizierung der Anforderungen und die Analyse der dafür geeigneten Technologien auf. Da wir unser Thema selbst eingebracht hatten, konnten wir die Anforderungen grösstenteils selbst bestimmen. Die Vorgaben besagten uns lediglich, dass unsere Applikation Java Server Faces 2.0 verwenden muss. Dies beschränkte uns in der Wahl der Technologie auf die Welt des Java EE. Nach einer ausführlichen Studie der einzelnen Typen von Applikationsservern, haben wir uns für GlassFish entschieden. Zusätzlich habe ich noch die JPA Thematik vertieft.

Nach Abschluss der ersten Iteration der Phase Elaboration haben wir mit der Entwicklung des Architektur Prototypen begonnen. Dieser beinhaltete den grundsätzlichen Aufbau der Software. Er zeigte die Verbindung vom User Interface bis zur Datenbank. Bei dieser Arbeit entdeckten wir einzelne Probleme mit dem Aufbau unserer Projektumgebung. Wir benötigten einige Stunden Arbeit, bis beide ein funktionsfähiges Eclipse mit einem integrierten GlassFish Server benutzen konnten. Mit dem Architektur Prototyp wurde auch der grösste Teil des Data Access Layer fertiggestellt.

Nach den Osterferien begannen wir mit der Implementierung der Geschäftslogik und den User Interfaces. Hierbei tauchten unter Anderen die im Abschnitt *Technologische Aspekte* beschriebenen Probleme auf. Diese konnten wir jedoch alle lösen und konnten einen stabilen Final Release erstellen, welche die Technologien, JSF, Primefaces, EJB und JPA vereint.

Da ich in der zweitletzten Woche am CCNA Kurs teilnahm, habe ich meinem Projektpartner einen Mehraufwand beschert, welchen er aber hervorragend gemeistert hat. In dieser Woche konnte ich lediglich eine kontrollierende Funktion einnehmen.

1.2 Rückblick

Aufgrund der gemeinsamen Projekterfahrung konnten wir unsere Stärken optimal einsetzen. Wodurch wir speditive und qualitativ hochstehende Leistungen erbringen konnten. Durch die gegenseitige Kontrolle wurde dies auch stets überprüft. An diesem Punkt möchte ich auch unserem Betreuer Prof. Hans Rudin für seine Unterstützung danken. Er betrachtete unsere Fortschritte stets mit einem kritischen Auge und konnte uns so auf mögliche Probleme und Möglichkeiten zu Verbesserungen hinweisen.

Mit den verschiedenen Technologien zu arbeiten war sehr spannend aber auch fordernd. Umso wichtiger war eine gute Studie der einzelnen Aspekte.

Die Verwendung des GlassFish Servers hat mich überzeugt. Ich werde ihn für weitere Projekte wiederverwenden und auch empfehlen. Er ist einfach zu konfigurieren und zu bedienen.

1.3 Lessons Learned

Mein Lernfortschritt über das Studium hinweg hat sich in dieser Arbeit klar aufgezeigt. Ich bin vom Nicht-Programmierer zum Junior Entwickler aufgestiegen. Auch konnte ich bei dieser Arbeit vieles dazu lernen. Beispielsweise die praktische Verwendung von JPA oder die Implementierung des Session Handlings.

Wie bereits angesprochen war ich in der zweitletzten Woche des Projektes aufgrund des CCNA-Kurses nicht verfügbar. Zudem hatte ich in der letzten Woche noch die Zertifikatsprüfung zu absolvieren. Dadurch ging uns einige kostbare Zeit verloren. Bei einer nächsten vergleichbaren Arbeit werde ich versuchen ein solches Szenario zu vermeiden.

2. Christian Zurbuchen

2.1 Projektverlauf

Die konkrete Aufgabe war zu Beginn des Projektes noch nicht ganz klar. Was uns inhaltlich erwarten würde wussten wir, da wir diese Arbeit persönlich einbrachten, jedoch waren die Details des Lieferumfanges noch nicht geklärt. Diese wurden in den ersten Wochen der Analyse und mittels der Entwicklung der Anforderungen von uns festgelegt und mit dem Betreuer abgesprochen. Als nun die Planung des Projektes definiert und die Anforderungen spezifiziert waren konnte das Projekt so richtig beginnen.

Da wir in unserem Projekt einen Applikationsserver verwendeten begannen wir zuerst mit der Suche nach möglichen Technologien und deren Analyse. Die engere Auswahl fiel auf JBoss und GlassFish, wobei wir uns dann für die Variante eines GlassFish Servers entschieden. Da dieser kritische Faktor des Projektes nun geklärt wurde konnte mit der Analyse der Problemdomain und der Weiterentwicklung der Anforderungen fortgefahren werden.

Gegen Mitte der Elaboration Phase begannen wir mit der Entwicklung eines Architektur Prototypen, welcher die Verbindung eines einfachen User Interfaces zur Datenbank in beide Richtungen aufzeigen sollte. Diese Entwicklung diente ebenfalls dazu um festzustellen, dass die Entscheidung für die Verwendung eines GlassFish Servers korrekt und vertretbar war. Es tauchten nach ersten Schwierigkeiten beim Setup der Entwicklungsumgebung keine weiteren Probleme im Zusammenhang mit dem GlassFish Server mehr auf.

Anschliessend begannen wir mit dem Design der Software und deren Architektur und fuhren mit der Entwicklung eines ersten GUI Prototyps fort. Dieser weitere Prototyp stellte einen bereits vollkommen funktionierenden Teilbereich dar. Die komplexeren User Interfaces wurden erst zu einem späteren Zeitpunkt entwickelt.

Auf die Entwicklung des GUI Prototyps folgte die Entwicklung des Beta und Final Releases. Die Aufwände für die Implementierung wurden stark unterschätzt. Schlussendlich verbrauchte die Entwicklung beinahe die Hälfte des kompletten Projektaufwandes.

Die Arbeit wurde uns etwas erschwert, da unser Betreuer in den letzten beiden Wochen vor der Abgabe nicht anwesend war. Strukturelle Aspekte des Hauptdokumentes zur Abgabe konnten vorab bereits besprochen werden, den inhaltlichen Bereich mussten wir jedoch ohne Rückmeldungen durchführen. Mit diesem Umstand wussten wir jedoch gut umzugehen und konnten diverse kritische Entscheidungen unter uns sinnvoll und leicht klären.

Die letzten beiden Wochen waren jedoch trotzdem eine stressige Angelegenheit, da man sich dauernd fragt, was man noch alles vergessen hat. Die Abwesenheit des Betreuers brachte zudem noch zusätzliche Unsicherheit bezüglich der Abgaben und der verlangten Inhalte. Wenn man dann schliesslich zum Ende kommt, sieht alles nur noch halb so schlimm aus.

2.2 Rückblick

Da wir nun bereits zum dritten Mal, nach dem Software Engineering 2 Projekt und der Semesterarbeit, zusammengearbeitet haben wurde eine sinnvolle Aufteilung der einzelnen Verantwortlichkeiten massiv vereinfacht. Ausserdem kennt man das Projektteam bereits gut und weiss somit in welchen Bereichen der Andere einen guten Beitrag leisten kann und in welchen die eigenen Stärken liegen. Durch diese Aufteilung und Kenntnis kamen wir auch mit der Einhaltung des Zeitplans gut zurecht.

Die Arbeit mit Redmine als Projektmanagementsoftware fand ich extrem sinnvoll. Der Umgang mit dem Tool ist recht einfach und wenn man sich daran gewöhnt hat, kann man ohne grosse Mühe einzelne Iterationen und Arbeitspakete planen. Auch die Erfassung der eigenen Arbeitszeit ist simpel gelöst und kann durch verschiedene Auswertungen beobachtet werden.

2.3 Lessons Learned

Durch die Durchführung des dritten Projektes an dieser Schule, merke ich, dass ich meine persönlichen Leistungen bezüglich des zeitlichen Aufwandes immer besser einschätzen kann. Wenn ich feststelle, dass wir noch folgendes Dokument erstellen müssen oder einen spezifischen Bereich in der Software entwickeln müssen, kann ich ziemlich genau abschätzen, wie viel Zeit ich dafür benötigen werde. Dies hilft mir natürlich in zukünftigen Projekten innerhalb meines Arbeitsumfeldes.

Wir haben uns für eine Entwicklung von Software mit dem Test Driven Development Ansatz entschieden. Dieser konnte zu Beginn gut angewendet werden, je mehr Zeitdruck jedoch entstand und sobald Probleme mit Klassen auftauchten, welche schwer zu testen sind, wurde dieser Ansatz leider etwas vernachlässigt. Im speziellen tauchten Schwierigkeiten beim Testen von Managed Beans auf, da interessante Methoden nur mit mühsamem Mocking testbar waren. Wir haben uns schliesslich dazu entschieden diese Beans sauber mittels der Systemtests abzudecken. Bei zukünftigen Projekten in meinem Arbeitsumfeld möchte ich den Test Driven Development Ansatz weiter und intensiver verwenden, da er viele Fehler der Software bereits vor der Entwicklung des Fehlers eliminiert. Ausserdem wird der Aufwand des Testens am Ende der Entwicklung massiv reduziert und die zukünftige Wartbarkeit der Software verbessert sich ebenfalls.

Durch die geplante und bekannte Abwesenheit unseres Betreuers kamen wir gegen Ende des Semesters etwas ins Schwitzen mit der Fertigstellung der strukturellen Inhalte der abzugebenden Dokumente. Dies führte in den kommenden letzten zwei Wochen zu Stress und vielen Unklarheiten. Wenn ich in Zukunft von solchen Einschränkungen weiss, werde ich die relevanten und wichtigen Abgaben noch frühzeitiger vor der Abwesenheit der beurteilenden oder leitenden Person klären. Somit bleibt für eine allfällige Rücksprache mehr Zeit und es können mehr Details genauer definiert werden.

Anhang A

1. Glossar

Begriff	Beschreibung
Ajax	Asynchronous JavaScript and XML; ermöglich asynchrone Datenübertragung zwischen Webbrowser und Server
Ant	Werkzeug zum automatisierten Erzeugen von ausführbaren Programmen aus dem Quellcode
Apache	ehrenamtlich arbeitende Organisation für Open Source Software; in diesem Fall Webserver der Apache Software Foundation
Assessment	engl. für Bewertung, Beurteilung
Beta Release	Unfertige Version einer Software; Meist Verwendung zu Testzwecken
BLL	Business Logic Layer (Geschäftslogikschicht); stellt die Schnittstelle für die grafische Benutzeroberfläche zum DAL dar
CLI	Command Line Interface
DAL	Data Access Layer (Datenzugriffsschicht); beinhaltet die Schnittstellen zur Datenbank mit den nötigen Abfragen
EclEmma	Eclipse Plug-In zur Analyse der Testabdeckung einer Software
Eclipse	Eclipse ist ein kostenfreies Programmierwerkzeug zur Entwicklung von Software verschiedenster Art
EJB	Enterprise Java Beans; standardisierte Komponenten innerhalb eines Java EE Servers
Emma	Freies Werkzeug zur Messung der Testabdeckung einer Java Anwendung
Entity	Modellieren die persistenten Daten des Systems
Final Release	Fertige / Endgültige Version einer Software
FindBugs	Eclipse Plug-In zur Analyse des Source Codes auf gewisse Mängel
GlassFish	Open Source Projekt eines Java EE Servers von Sun Microsystems
GUI	Grafisches User Interface; grafische Benutzeroberfläche einer Software
Guideline	engl. für Richtlinie
Hibernate	Open Source Persistenz Framework für Java von Red Hat
HSR	Hochschule für Technik Rapperswil
ISO	International Organization for Standardization (Internationale Organisation für Normung); erarbeitet internationale Normen
Java EE	Java Enterprise Edition; Webapplikationen in Java
Javadoc	Software-Dokumentationswerkzeug, das aus Java-Quelltexten automatisch HTML-Dokumentationsdateien erstellt
JBoss	Open Source Projekt eines Java EE Servers von Red Hat
Jenkins	Software für die kontinuierliche Integration in Softwareprojekten

Begriff	Beschreibung
JMock	Programmbibliothek zur Erstellung von Mock Objekten für Java
JPA	Java Persistence API (Application Programming Interface); Schnittstelle für Java Anwendung zur Übertragung von Objekten in Datenbankeinträge
JRE	Java Runtime Environment
JSF	Java Server Faces; Framework Standard zur Entwicklung von grafischen Benutzeroberflächen für Webapplikationen
Kohäsion	Beschreibt, ob eine logische Einheit oder Aufgabe sinnvoll abgebildet ist
Komposition	Starke Aggregation (Beziehung) innerhalb UML Diagramm; die Existenz des einen Objektes ist von der Existenz des Anderen abhängig
Lost Update	Fehler der bei parallelen Schreibzugriffen auf dasselbe Objekt auftauchen kann
Managed Bean	Bean Objekte die vom Framework JSF verwaltet werden
Metrik	Mathematische Funktion, welche die Eigenschaft von Software als Zahlenwert abbildet
Mockup, Mocking	engl. für Attrappe; in diesem Fall Attrappe von möglichen grafischen Benutzeroberflächen oder einer Datenbank
Open Source	Software, deren Quelltext öffentlich zugänglich ist
PM	Projektmanagement
Primefaces	Komponentenbibliothek zur Unterstützung / Erweiterung von JSF
Prototyp	Unfertige Version einer Software; Dient zur Veranschaulichung / Testen einer gewissen Komponente der zukünftigen Software
Redmine	Open Source Projektmanagementsoftware
RUP	Rational Unified Process; Vorgehensmodell zur Software Entwicklung
SAD	Software Architektur Dokument; Beschreibt das Zusammenspiel der einzelnen Komponenten innerhalb eines Softwaresystems.
SE	Software Engineering
SQL	Datenbanksprache zur Definition von Datenstrukturen in relationalen Datenbanken sowie zum Bearbeiten und Abfragen von darauf basierten Datenbeständen
SSH	Secure Shell; Netzwerkprotokoll mit verschlüsselter Übertragung
STAN	Eclipse Plug-In zur Analyse des strukturellen Aufbaus einer Software
SVN	Apache Subversion; Freie Software zur Versionsverwaltung von Dateien und Verzeichnissen
teamMgmt	Abkürzung für Team Management (steht in diesem Fall für Verwaltung von Mitarbeitern und Abwesenheiten)
Test Driven Development	Methode zur agilen Software Entwicklung; Programmierer erstellt Tests konsequent vor der zu testenden Komponente
UML	Unified Markup Language; ermöglicht die grafische Modellierung von Software
XHTML	Extensible Hypertext Markup Language; textbasierte Auszeichnungssprache

Tabelle 70: Glossar

2. Verzeichnisse

2.1 Literatur

- Java Server Faces 2.0, Grundlagen und erweiterte Konzepte; Martin Marinschek, Michael Kunz, Gerald Müllan; 2. Auflage; ISBN 978-3-89864-606-2
- JSF 2.0 Online Referenz
  http://jsfatwork.irian.at/semistatic/introduction.html (Stand 12.06.2012)
- Primefaces Showcase
  http://www.primefaces.org/showcase/ui/home.jsf (Stand 12.06.2012)
- GlassFish Quickstart Guide
  http://glassfish.java.net/docs/3.1.1/quick-start-guide.pdf (Stand 26.05.2012)
- Norm ISO 9126
  http://de.wikipedia.org/wiki/ISO/IEC_9126 (Stand 10.03.2012
- Vorlesungsunterlagen Software Engineering 2 (Stand 2011)
- Vorlesungsunterlagen Enterprise Computing (Stand 2011)
- Applikationsserver Vergleich
  http://www.computerwoche.de/software/software-infrastruktur/1873146/ (Stand 07.03.2012)
- Vergleich GlassFish und JBoss
  http://www.oracle.com/us/products/middleware/application-server/oracle-glassfish-server/comparing-glassfish-jboss-wp-117118.pdf (Stand 07.03.2012)
- Vergleich GlassFish Version 2 und Version 3
  http://glassfish.java.net/public/comparing_v2_and_v3.html (Stand 09.03.2012)
- Dokumentation JBoss
  https://docs.jboss.org/author/display/AS71/Documentation ff (09.03.2012)

2.2 Tabellen

Tabelle 1: Anforderungsspezifikation: Allgemeine Anforderungen 19
Tabelle 2: Anforderungsspezifikation: Anforderungen Bereich Mitarbeiter (Abteilungen, Projekte) .. 20
Tabelle 3: Anforderungsspezifikation: Anforderungen Bereich Absenzen 20
Tabelle 4: Anforderungsspezifikation: Rollenrechte Bereich Mitarbeiter 23
Tabelle 5: Anforderungsspezifikation: Rollenrechte Bereich Abteilungen & Projekte 23
Tabelle 6: Anforderungsspezifikation: Rollenrechte Bereich Absenzen 23
Tabelle 7: Anforderungsspezifikation: UC1: Login 24
Tabelle 8: Anforderungsspezifikation: UC2: Change password 24
Tabelle 9: Anforderungsspezifikation: UC3: CRUD employee 24
Tabelle 10: Anforderungsspezifikation: UC4: CRUD absence 24
Tabelle 11: Anforderungsspezifikation: UC5: Approve/Disapprove absence 25
Tabelle 12: Anforderungsspezifikation: UC6: CRUD department 25
Tabelle 13: Anforderungsspezifikation: UC7: CRUD project 25
Tabelle 14: Anforderungsspezifikation: UC8: Manage absence types 26
Tabelle 15: Domainanalyse: Employee Attribute 31
Tabelle 16: Domainanalyse: Employee Beziehungen 32
Tabelle 17: Domainanalyse: Department Attribute 32
Tabelle 18: Domainanalyse: Department Beziehungen 32
Tabelle 19: Domainanalyse: Project Attribute 33
Tabelle 20: Domainanalyse: Project Beziehungen 33
Tabelle 21: Domainanalyse: Worker Attribute 33
Tabelle 22: Domainanalyse: Worker Beziehungen 33
Tabelle 23: Domainanalyse: Absence Attribute 34
Tabelle 24: Domainanalyse: Absence Beziehungen 34
Tabelle 25: Domainanalyse: AbsenceType Attribute 34
Tabelle 26: Domainanalyse: AbsenceType Beziehungen 34
Tabelle 27: Domainanalyse: Contract CO1: login 38
Tabelle 28: Domainanalyse: Contract CO2: changePassword 38
Tabelle 29: Domainanalyse: Contract CO3: createEmployee 38
Tabelle 30: Domainanalyse: Contract CO4-1: createAbsence 39
Tabelle 31: Domainanalyse: Contract CO4-2: createAbsence 39
Tabelle 32: Domainanalyse: Contract CO5: setAbsenceState 39
Tabelle 33: Domainanalyse: Contract CO6: createDepartment 40
Tabelle 34: Domainanalyse: Contract CO7: createProject 40
Tabelle 35: Domainanalyse: Contract CO8: manageAbsenceTypes 40
Tabelle 36: Technischer Bericht: JBoss Funktionalitäten 42
Tabelle 37: Technischer Bericht: GlassFish Funktionalitäten 43
Tabelle 38: Technischer Bericht: Bewertungstabelle 45
Tabelle 39: Software Architektur Dokument: WebContent Ordner 61
Tabelle 40: Software Architektur Dokument: bean Subpackages und Klassen 65
Tabelle 41: Software Architektur Dokument: domain Subpackages und Klassen 66
Tabelle 42: Software Architektur Dokument: handler Klassen 67
Tabelle 43: Software Architektur Dokument: service Klassen 70
Tabelle 44: Software Architektur Dokument: util Klassen 73
Tabelle 45: Handbuch: Standardeinstellungen 80
Tabelle 46: Projektplan: Organisationsstruktur 84
Tabelle 47: Projektplan: Externe Schnittstellen 84
Tabelle 48: Projektplan: Qualitätsmassnahmen 91
Tabelle 49: Systemtestspezifikation: Test (Login & Logout) 97
Tabelle 50: Systemtestspezifikation: Test (Passwort ändern) 98

Tabelle 51: Systemtestspezifikation: Test (Absenztypen verwalten) 99
Tabelle 52: Systemtestspezifikation: Test (Neuer Mitarbeiter) 100
Tabelle 53: Systemtestspezifikation: Test (Meine Daten) 101
Tabelle 54: Systemtestspezifikation: Test (Mitarbeiter suchen) 102
Tabelle 55: Systemtestspezifikation: Test (Neue Abteilung) 103
Tabelle 56: Systemtestspezifikation: Test (Abteilung suchen) 104
Tabelle 57: Systemtestspezifikation: Test (Neues Projekt) 105
Tabelle 58: Systemtestspezifikation: Test (Meine Projekte) 106
Tabelle 59: Systemtestspezifikation: Test (Projekt suchen) 107
Tabelle 60: Systemtestspezifikation: Test (Neue Absenz, einmalig) 108
Tabelle 61: Systemtestspezifikation: Test (Neue Absenz, periodisch) 109
Tabelle 62: Systemtestspezifikation: Test (Meine Absenzen, einmalig) 110
Tabelle 63: Systemtestspezifikation: Test (Meine Absenzen, periodisch) 111
Tabelle 64: Systemtestspezifikation: Test (Genehmigung) 112
Tabelle 65: Systemtestspezifikation: Test (Übersicht) 113
Tabelle 66: Systemtestspezifikation: Test (Rollen & Rechte, Mitarbeiter) 114
Tabelle 67: Systemtestspezifikation: Test (Rollen & Rechte, Abteilung) 114
Tabelle 68: Systemtestspezifikation: Test (Rollen & Rechte, Projekte) 115
Tabelle 69: Systemtestspezifikation: Test (Rollen & Rechte, Absenzen) 115
Tabelle 70: Glossar 130

2.3 Abbildungen

Abbildung 1: Management Summary: Systemarchitektur 14
Abbildung 2: Management Summary: Übersicht der Absenzen 16
Abbildung 3: Anforderungsspezifikation: Use Case Diagramm 22
Abbildung 4: Domainanalyse: Domainmodell 30
Abbildung 5: Domainanalyse: SSD UC1: Login 35
Abbildung 6: Domainanalyse: SSD UC2: Change Password 35
Abbildung 7: Domainanalyse: SSD UC3: CRUD Employee 35
Abbildung 8: Domainanalyse: SSD UC4: CRUD absence 36
Abbildung 9: Domainanalyse: SSD UC5: Approve/Disapprove absence 36
Abbildung 10: Domainanalyse: SSD UC6: CRUD department 36
Abbildung 11: Domainanalyse: SSD UC7: CRUD project 37
Abbildung 12: Domainanalyse: SSD UC8: Manage absence types 37
Abbildung 13: Technischer Bericht: JSF Template (1/2) 46
Abbildung 14: Technischer Bericht: JSF Template (2/2) 46
Abbildung 15: Technischer Bericht: JSF Template Alternative (1/3) 47
Abbildung 16: Technischer Bericht: JSF Template Alternative (2/3) 47
Abbildung 17: Technischer Bericht: JSF Template Alternative (3/3) 48
Abbildung 18: Technischer Bericht: JSF Lebenszyklus 48
Abbildung 19: Technischer Bericht: Doppelklick Alternative (1/2) 49
Abbildung 20: Technischer Bericht: Doppelklick Alternative (2/2) 49
Abbildung 21: Technischer Bericht: Schedule Komponente Alternative (1/2) 50
Abbildung 22: Technischer Bericht: Schedule Komponente Alternative (2/2) 50
Abbildung 23: Technischer Bericht: EntityManager 51
Abbildung 24: Technischer Bericht: Rollen & Rechte Alternative (1/2) 52
Abbildung 25: Technischer Bericht: Rollen & Rechte Alternative (2/2) 53
Abbildung 26: Technischer Bericht: Konfigurierbare Absenztypen 54
Abbildung 27: Software Architektur Dokument: Systemarchitektur 56
Abbildung 28: Software Architektur Dokument: Klasse Employee 59
Abbildung 29: Software Architektur Dokument: Logische Architektur 61

Abbildung 30: Software Architektur Dokument: WebContent 61
Abbildung 31: Software Architektur Dokument: Aufbau der Hauptseite 62
Abbildung 32: Software Architektur Dokument: Geschäftslogik 63
Abbildung 33: Software Architektur Dokument: Package bean 64
Abbildung 34: Software Architektur Dokument: Package domain 66
Abbildung 35: Software Architektur Dokument: Package handler 67
Abbildung 36: Software Architektur Dokument: Package exception 68
Abbildung 37: Software Architektur Dokument: Datenzugriffsschicht 69
Abbildung 38: Software Architektur Dokument: Package service 70
Abbildung 39: Software Architektur Dokument: Package entity 71
Abbildung 40: Software Architektur Dokument: Klasse EntityBase 72
Abbildung 41: Software Architektur Dokument: Package exception 72
Abbildung 42: Software Architektur Dokument: Package util 73
Abbildung 43: Software Architektur Dokument: Package übergreifende Abhängigkeiten 74
Abbildung 44: Software Architektur Dokument: Sequenzdiagramm 75
Abbildung 45: Software Architektur Dokument: Datenmodell 77
Abbildung 46: Projektplan: Überblick Phasen, Iterationen und Meilensteine 85
Abbildung 47: Zeitauswertung: Auswertung Disziplinen 95
Abbildung 48: Zeitauswertung: Auswertung Iterationen/Disziplinen 96
Abbildung 49: Zeitauswertung: Auswertung Mitglieder 96
Abbildung 50: Qualitätssicherung: Testabdeckung (alle Packages) 116
Abbildung 51: Qualitätssicherung: Testabdeckung (ohne Package bll.bean) 116
Abbildung 52: Qualitätssicherung: Struktur Software 117
Abbildung 53: Qualitätssicherung: Struktur Package bll 117
Abbildung 54: Qualitätssicherung: Struktur Package bll.bean 118
Abbildung 55: Qualitätssicherung: Struktur Package dal 118
Abbildung 56: Qualitätssicherung: Cyclomatic Complexity 119
Abbildung 57: Qualitätssicherung: Feature Envy 120
Abbildung 58: Qualitätssicherung: Lines of Code in Method 121
Abbildung 59: Qualitätssicherung: Efferent Coupling 122
Abbildung 60: Qualitätssicherung: Lack of Cohesion 123

Anhang B

Auf folgende Anhänge wird verwiesen, die sich auf der abgegebenen CD befinden:

- **Iterationspläne & -assessments** - Ordner doc/02_Iterationsplanung
- **Systemtestprotokolle** - Ordern doc/14_Qualitätsmanagement/Systemtestprotokolle
- **Metriken** - Ordner doc/14_Qualitätsmanagement/Metriken/index.html
- **Sitzungsprotokolle** - Ordner doc/03_Protokolle

- **Vereinbarung der Rechte** - Ordner doc/Vereinbarung.jpeg

- **Source Code** - Ordner src
- **Javadoc** - Ordner src/doc