Bachelorarbeit

Mandate based IP-Telephony Administration (Mabata)

Modul:	Bachelorarbeit FS09	
Projektteam:	Fabio Looser	f1looser@hsr.ch
	Ueli Kuratli	ukuratli@hsr.ch
	Pascal Fuchs	pfuchs@hsr.ch
Betreuer:	René Stanger	rstanger@hsr.ch
	Maurin Egler	megler@hsr.ch
Verantwortlicher:	Prof. Beat Stettler	bstettle@hsr.ch
Datum:	12. Juni 2009	

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

ins institute for networked solutions

Abstract

Der Flughafenbetreiber Unique stellt seinen Kunden (Fluggesellschaften, Detailhandelsgeschäften, Restaurants, …) diverse Services zur Verfügung. Unter anderem betreibt Unique eine Telekommunikationsinfrastruktur, welche von ihren Kunden genutzt wird. Das vorhandene Telefonsystem möchte Unique mit der VoIP-Lösung von Cisco, dem Cisco Unified Communications Manager (CUCM) ablösen. Dieser ist jedoch nicht mandantenfähig, weshalb Unique alle administrativen Aufgaben wie z.B. Benutzer erfassen, Telefonnummern ändern, usw. selber erledigen müsste. Aus dieser Problemstellung entstand das Projekt Mabata, welches den CUCM mandantenfähig macht und den Kunden die Möglichkeit gibt, deren Einstellungen selbst zu verwalten.

Der grösste Teil dieser Arbeit besteht darin, eine neue Schicht zwischen Mabata und dem CUCM zu entwickeln. Ziel dieser Middleware ist, dass zukünftig mehrere Applikationen über diese Middleware an den CUCM angeschlossen werden können. Dadurch muss die Anbindung nur einmal programmiert werden. Ein weiterer Schritt besteht darin, weitere Zielsysteme an die Middleware anzuschliessen. Damit wird die Middleware die zentrale Software zwischen Applikationen und Zielsystemen. In dieser Arbeit beschränkt sich die Entwicklung der Middleware darauf, die Kommunikation zwischen Mabata und dem CUCM zu unterstützen.

Ein anderer grosser Punkt dieser Arbeit besteht darin, die Benutzeroberfläche zu vereinfachen. In der aktuellen Version gleicht diese der komplexen Umgebung des CUCM. Das neue GUI soll diesen möglichst stark abstrahieren. Als Resultat dieser neuen Benutzeroberfläche kann ein Mandant seine Arbeit mit weniger Aufwand, dafür mit grösserer Übersicht, erledigen.

Das Ergebnis dieser Arbeit besteht einerseits aus der Software Mabata, welche in einer produktiven Cisco VoIP-Lösung einen erheblichen Vorteil in der Administration des CUCM mit sich bringt, und andererseits ist eine Middleware entstanden, welche eine gute Grundlage für Weiterentwicklungen bietet.

Dokumente des Projekts

Die Struktur der Bachelorarbeitsdokumentation wurde in zwei Dokumente gegliedert:

Bachelorarbeit – *aktuelles Dokument*

Das ist das Hauptdokument der Bachelorarbeit. Darin enthalten sind:

- Auftrag
- Management Summary
- Technischer Bericht
- Projektplan
- Projektmonitoring

Technische Referenz

Dokument für die Weiterentwicklung und interessierte Spezialisten. Darin enthalten sind:

- Entwicklungsumgebung
- Installationsanleitung
- Analyse
- Anforderungsspezifikation
- Design

Inhaltsverzeichnis

1. Auftrag

1.1. Aufgabenstellung

Diese Bachelorarbeit basiert auf vorgängigen Arbeiten der Applikation Mabata. In den Vorarbeiten wurde die mandantenfähige Webapplikation entwickelt, welche die Administration des Communications Managers abbildet.

In dieser Bachelorarbeit gib es zwei Schwerpunkte. Diese bestehen einerseits aus einer Vereinfachung der bestehenden Mabata Benutzeroberfläche. Dadurch soll erreicht werden, dass Mabata nicht mehr die komplexen Abläufe vom Communications Manager abbildet, sondern diese abstrahiert und vereinfacht.
Der andere Schwerpunkt ist die Entwicklung einer neuen Middleware. Diese soll in einem ersten Stadium dazu dienen, die Kommunikation zwischen Mabata und dem Communications Manager zu ermöglichen. Dabei ist die Middleware in der Lage, die gesamten von Mabata abstrahierten Business Cases abzuarbeiten.

Neben diesen beiden Schwerpunkten wird Mabata um weitere Funktionen erweitert. Durch ein intensives Testen soll die Stabilität deutlich verbessert werden.

1.2. Analyse

Da sich das gesamte Team bereits in der Studienarbeit ausgiebig mit Mabata auseinandergesetzt hat, musste die Applikation nicht mehr analysiert werden. Aus diesem Grund konnte das Augenmerk von Anfang an auf die Analyse der neuen Funktionen gelegt werden.

1.3. Design

Die Architektur von Mabata soll belassen werden wie bisher. Für die Vereinfachung der Benutzeransicht genügt, das GUI und die Controller anzupassen, wobei dies an der aktuellen Architektur nichts ändert. Für die neu zu implementierenden Funktionen muss jeweils ein neues Design für die Umsetzung erstellt werden.

Da die Middleware neu zu implementieren ist, wird die ganze Architektur von Grund auf neu entworfen. Dabei werden verschiedene Designüberlegungen miteinander nach deren Vor- und Nachteilen verglichen und die am besten geeignete Variante weiter ausgearbeitet.

1.4. Präsentation & Dokumentation

Das Projekt soll gut dokumentiert werden, so dass die Weiterentwicklung einem anderen Team (Studienarbeit oder Bachelorarbeit) erleichtert wird. Ausserdem soll eine Anleitung für den zukünftigen Administrator von Mabata entstehen, welche ihn durch die Installation leitet.

Die erarbeiteten Lösungen werden am Schluss der Bachelorarbeit in einer mündlichen Prüfung präsentiert.

1.5. Folgearbeit

Da es sich bei dieser Bachelorarbeit um eine Folgearbeit handelt, ist in diesem Abschnitt aufgeführt, was in den Vorarbeiten und was im Rahmen dieser Bachelorarbeit erarbeitet wurde. Die Verwendete Abkürzung CRUD steht für Create, Read, Update und Delete.

1.5.1. Bestehende Funktionen welche nicht verändert wurden

Funktion	Beschreibung
Funktionen des Administrators	
Authentifizierung	CRUD Radiusserver
Inventar	Neue Telefone können erfasst oder per Inventardatei eingelesen werden
Portierung	Erlaubt Benutzer von Mandant A nach Mandant B zu portieren
Statistik	Hier wird für jeden Mandant aufgelistet, wie viele aktiven/inaktiven Telefone, Profile und aktive/inaktive Nummern er besitzt
Funktionen des Superusers	
Statistik	Unter diesem Menüpunkt wird eine Übersicht über die Telefonnummern, Telefone, Profile und Benutzer des Mandanten gegeben
Pickup Gruppen	CRUD Pickup Gruppen
Hunt Gruppen	CRUD Hunt Gruppen
Funktionen des Users	
PIN	Ändern des Pins
Kurzwahl	Erfassen und Ändern der Kurzwahltasten
Weiterleitung	CRUD Weiterleitungen

Tabelle 1 Bestehende Funktionen welche nicht verändert wurden

1.5.2. Bestehende Funktionen welche verändert wurden

Funktion	Beschreibung
Funktionen im Core	
myFramework	Aus Mabata entfernt und in Middleware integriert
Navigation	Navigation verbessert
Funktionen des Administrators	
Mandator	Nummernblock und Inventar in CRUD Mandator integriert
Importassistent	Performance optimiert, Pickup Gruppen Import verbessert, Hunt Gruppen Import hinzugefügt
Konsistenzprüfung	Performance optimiert, Hunt Gruppen hinzugefügt
Reporting	Reportingfunktion komplett neu erstellt, inkl. Syslog-Anbindung
Funktionen des Superusers	
Benutzer	Telefon und Profil in CRUD Benutzer integriert
Nummern	Hunt- und Pickup Gruppen-Nummern werden angezeigt

Tabelle 2 Bestehende Funktionen welche verändert wurden

1.5.3. Neue Funktionen

Funktion	Beschreibung
Funktionen im Core	
Nummernpräfix	Unterstützung von Nummernpräfix für Communications Manager 7.x integriert
getException	Fehlermeldung, falls Objekt auf Communications Manager nicht mehr existiert
Umlaute	Unterstützung von Umlauten in myFramework integriert
Passwort-Hash	Nur Hash-Werte von Passwörter in Datenbank abspeichern
Softphones	Softphones werden von Mabata unterstützt
Funktionen des Superusers	
Reporting	Übersicht der durchgeführten Aktionen
SoftKey Template	SoftKey Templates können den Telefonen/Profilen zugewiesen werden
Funktionen des Users	
Passwort	Benutzer können ihr eigenes Passwort ändern

Tabelle 3 Neue Funktionen

Zusätzlich zu den Mabata Funktionen wurde eine Middleware erstellt, über welche die Kommunikation zwischen Mabata und dem Cisco Communications Manager stattfindet.

2. Management Summary

2.1. Motivation und Ausgangslage

Cisco hat mit dem Communications Manager eine digitale Telefonzentrale für Grossunternehmen entwickelt. Diese ist für den Betrieb von einigen hundert bis mehreren tausend Voice over IP Telefone gedacht. Die Administration geschieht über ein aufwändiges und kompliziertes Web-Interface. Ausserdem können die meisten Funktionen auch über eine zusätzliche Service-Schnittstelle angesteuert werden.

Cisco hat in ihrem Produkt ein Berechtigungssystem eingeführt, in welchem verschiedene Aufgaben an einzelne Administratoren delegiert werden können. Dies bedeutet beispielsweise, dass die Verwaltung von Telefonnummern und Telefonen getrennt werden kann. Dieses Berechtigungssystem ist jedoch für die Firma Unique ungenügend.

Mabata ist eine Administrations-Applikation für den Communications Manager. Sie erweitert seine Möglichkeiten in Bezug auf die Verwaltung und abstrahiert die bestehenden Funktionen, dass diese von der jeweils dafür verantwortlichen Person ausgeführt werden können.

In dieser Bachelorarbeit geht es darum, nebst dem Vorteil welchen Mabata durch das Berechtigungssystem mit sich bringt, auch eine Vereinfachung im Ablauf der Konfiguration einzubauen. Dabei wird die gesamte Benutzeroberfläche von Mabata analysiert und wo nötig die Abläufe vereinfacht.

Um das gewonnene Wissen über die Kommunikation mit dem Communications Manager nicht nur mit Mabata verwenden zu können, wird eine neue Zwischenschicht entwickelt, welche später von verschieden Applikation verwendet werden kann.

Am Ende dieser Arbeit soll ein Gesamtprodukt entstehen, welches in einer produktiven Umgebung eingesetzt und dadurch einen entscheidenden Vorteil erzielt werden kann.

2.2. Vorgehen

Da sich alle Teammitglieder bereits in der Studienarbeit mit Mabata befasst haben, konnte am ersten Tag der Arbeit die Entwicklungsumgebung eingerichtet werden und direkt in die Analysephase des Projektes eingestiegen werden.

In den ersten sechs Wochen wurde der Grundstein für den späteren Verlauf der Bachelorarbeit gelegt. In dieser Zeit wurde die gesamte Benutzeroberfläche analysiert und anhand von Paperprototypen mögliche Verbesserungen veranschaulicht. Neben der neuen Benutzeroberfläche wurde eine Architektur für die Middleware entwickelt. Dabei wurde jeweils wöchentlich eine Sitzung mit den Betreuern abgehalten, die neu entwickelten Lösungen und Ideen vorgestellt, diese diskutiert und jeweils auf die nächste Sitzung wieder die neuen Ideen und Anmerkungen der Gespräche in die Architektur eingebaut. Dabei entstand nach sechs Wochen ein gut ausgearbeitetes Design, welches für die Implementierung sehr hilfreich war.

Während den ersten sechs Wochen wurden auch Funktionen und Erweiterungen analysiert und ausgearbeitet, welche in der Implementierungsphase nicht umgesetzt wurden. Dies hatte unterschiedliche Gründe. Teilweise wurde erkannt, dass die zu untersuchende Funktion nicht umzusetzen ist, und teilweise musste aus Zeitmangel die weniger wichtigen Funktionen weggelassen werden.

Im Projektplan war eine Woche Osterferien eingeplant. Da aber Mabata kurz nach Projektabschluss in einer produktiven Umgebung installiert werden soll, wurde während dieser Woche gearbeitet, damit der Meilenstein 3, in welchem ein Prototyp abgeliefert wurde, vorverlegt werden konnte. Damit blieb mehr Zeit, diesen ausgiebig zu testen. Aufgrund dieses Umstandes wurde die Entwicklung des Services Frameworks von den Betreuern übernommen. Diese gewonnene Zeit konnte in Verbesserungen des Mabata Cores investiert werden. Dadurch konnte am Ende des Projekts ein Produkt abgegeben werden, welches sehr viel benutzerfreundlicher und stabiler funktionierte als zu Beginn der Arbeit.

2.3. Ergebnisse

Das Ergebnis dieser Bachelorarbeit ist eine stabile Web-Applikation, welche zum ersten Mal seit dem Anfang seiner Entwicklung in einer produktiven Umgebung eingesetzt werden kann. Für die Benutzer von Mabata als grösster Unterschied ersichtlich ist die überarbeitete Benutzeroberfläche. Die Art und Weise wurde belassen wie bisher, da das Aussehen überzeugen konnte. Allerdings wurden die zentralen Abläufe entscheidend vereinfacht. Ein Mandant kann geführt einen neuen Benutzer mit Telefon, Profilen und dazugehörigen Nummern konfigurieren.

Da das Services Framework nicht realisiert wurde, war gegen Ende des Projektes genügend Zeit vorhanden, um viele kleine Fehler und Unschönheiten in Mabata zu bereinigen. All diese Punkte sind im Einzelnen keine grosse Sache, doch sind sie nötig, um eine stabile und benutzerfreundliche Applikation an den Kunden zu liefern.

Die Middleware wurde von Grund auf neu erstellt und war die grosse Herausforderung dieser Arbeit. Dabei mussten viele Punkte berücksichtigt werden, die zwar nicht direkt Implementiert wurden, jedoch für die spätere Weiterentwicklung benötigt werden.

Während dem gesamten Projekt war das Testing ein zentrales Element. Dies hat mehrere Gründe. Zum einen wurden durch die Umstellung im GUI die gesamten Abläufe geändert, was ein komplettes Testen nach sich zog. Ein weiterer Grund war die Erstellung der Middleware. Da neu alle Aufrufe an den Communications Manager über die Middleware gesendet werden, mussten alle Aufrufe getestet werden. Zu guter Letzt musste Mabata in einer Laborumgebung auf Herz und Nieren geprüft werden, da einem Kunden eine fehlerhafte Software nicht zugemutet werden kann. Dank diesen vielen Tests konnten viele Fehler beseitig werden, welche nicht direkt mit der aktuellen Arbeit zu tun hatten. Z.B. konnte im myFramework, welches den Zugriff von der Middleware auf den Communications Manager herstellt, einige entscheidende Fehler wettgemacht werden.

Dank dieser sehr anspruchsvollen Bachelorarbeit, welche uns mit verschiedenen Aspekten beschäftigte, war natürlich der Lernerfolg umso höher. Mabata verwendet viele verschiedene Technologien, mit welchen wir uns alle beschäftigen konnten. Auch die Projektplanung musste sehr streng eingehalten werden, da die Software kurz nach der Bachelorarbeit produktiv geht. Dies gelang uns sehr gut.

2.4. Ausblick

In dieser Arbeit wurden viele Fehler in Mabata behoben. Diese Fehler konnten grösstenteils aufgrund der ausgiebigen Tests eruiert werden. Durch die Neugestaltung des Konfigurationsablaufes, kann Mabata auch von Benutzern verwaltet werden, welche sich mit dem Cisco Communications Manager nicht auskennen.

Mit der Middleware, welche zwischen Mabata und dem Communications Manager interagiert, wurde eine neue Applikation mit viel Potential geschaffen. Diese Middleware ist momentan einzig für Mabata und den Communications Manager ausgelegt. Eine Erweiterung, um mehrere Quell- und Zielsysteme über diese Middleware anzubinden, könnte sicherlich viele Vorteile schaffen. Dadurch könnte z.B. ein in einer Applikation erstellter Benutzer automatisch auf verschiedenen Serversystemen erstellt werden. Zudem würde die Middleware für Entwickler ein einziges Interface bieten, um mehrere Systeme anzusprechen.

Werden sehr viele Benutzer, Telefone oder Nummern in Mabata verwaltet, machen sich zum Teil Verzögerungen in der Benutzeroberfläche bemerkbar. Diese entstehen dadurch, dass bei jeder Ansicht die gesamten Daten geladen und angezeigt werden. Eine Aufteilung dieser Daten auf mehrere Seiten würde die Performance der Benutzeroberfläche verbessern. Desweitern könnte die Performance eventuell durch das Optimieren der Datenbank-Konfiguration positiv beeinflusst werden.

Die Kostenabrechnung wurde in dieser Arbeit analysiert und die Grundlagen für die Architektur ausgearbeitet. Vor der Umsetzung dieser Funktion müssten die genauen Anforderungen des Kunden spezifiziert und analysiert werden.

Die Implementation der Nummern in Mabata ist nicht optimal gelöst. Durch den Präfix wird zwar das «+» oder «00» vor der Nummer ermöglicht, jedoch wäre es sinnvoll, den gesamten Nummerntyp in Mabata auf String umzustellen. Dadurch könnten komplexe Nummern direkt abgebildet werden.

Wichtig für den Erfolg von Mabata ist mit Sicherheit, dass die noch auftretenden Fehler rasch behoben und somit der Betrieb sichergestellt wird. Zudem könnte Mabata über die Auswertung von Benutzer-Feedbacks verbessert und mit neuen Funktionen ergänzt werden.

3. Technischer Bericht

3.1. Einleitung und Übersicht

Der Flughafenbetreiber Unique möchte ihre bestehende Telefonanlage mittels des Cisco Communications Managers auf Voice over IP umstellen. Um den Communications Manager mandantenfähig zu machen, wurde die Software Mabata entwickelt. Zudem soll die Konfiguration in Mabata gegenüber dem Communications Manager vereinfacht werden.

Mabata ist das Resultat von mehreren Studien- und Diplomarbeiten. Es wurde ständig überarbeitet und weiterentwickelt. Zu Beginn dieser Bachelorarbeit funktionierten die Grundlegenden Operationen, welche für die Konfiguration einer Telefonumgebung benötigt werden. Die Mandantenfähigkeit, und somit einer der Gründe, weshalb Mabata entstanden ist, war demnach bereits umgesetzt. Die Vereinfachung der Konfigurationsabläufe wurde jedoch nicht zufriedenstellend umgesetzt. In Mabata wurden die Abläufe aus dem Communications Manager übernommen. Das heisst, jedes Objekt musste zuerst auf der entsprechenden Oberfläche erstellt werden, bevor es einem anderen Objekt zugewiesen werden konnte. Beim erstellen eines Benutzer mit einem Telefon musste beispielsweise zuerst das Telefon über das Telefon-Menü erstellt werden, bevor das Telefon dem Benutzer im Benutzer-Menü hinzugefügt werden konnte. Dieser Ablauf musste vereinfacht werden, damit Objekte inkl. deren Zusammenhänge in derselben Oberfläche erstellt werden können, ohne dass in die verschiedenen Menüs gewechselt werden muss.

Die Vereinfachungen der Arbeitsabschritte wurden mithilfe einer neuen Applikation umgesetzt. Diese neue Applikation entstand in dieser Bachelorarbeit. Ziel dieser Applikation ist es, als Middleware ein Serversystem verschiedenen Anwendungen bereitzustellen. Die Grundidee dieser Middleware entstand daraus, dass verschiedene Serversysteme von mehreren Anwendungen angesprochen werden können. Wird zum Beispiel ein Benutzer in einer Anwendung erstellt, wird dieser Benutzer von der Middleware automatisch auf allen Zielsystemen erstellt. In dieser Arbeit beschränkten sich die Anforderungen für die Middleware bezüglich Zielsysteme darauf, Mabata an den Communications Manager anzubinden.

In einer separaten Diplomarbeit entstand ein Services Framework, mit dessen Hilfe die Telefonservices vom Communications Manager mandantenfähig gemacht wurden. Dieses Framework wurde damals direkt in eine Mabata Version eingefügt. Infolge der starken Kopplung zwischen dieser Mabata Version und dem Services Framework konnte das Framework nicht in die aktuelle Mabata Version übernommen werden. Aus dieser Tatsache folgte eine weitere Aufgabe für diese Arbeit. Das Services Framework soll von Mabata herausgelöst und selbstständig lauffähig sein.

3.2. Ergebnisse

Das Erste und vom Endbenutzer am grössten wahrgenommene Ergebnis ist die erfolgreiche Umsetzung der Mabata Business Case Vereinfachung. Vor der Umsetzung musste ein Benutzer, welcher Mabata bedient, wissen, wie der Communications Manager genau konfiguriert werden muss. Die Abfolge der einzelnen Schritte der Konfiguration mussten eingehalten werden. Zum Beispiel musste zuerst eine Nummer, dann ein Telefon und erst zum Schluss der Benutzer selber eingerichtet werden. Mit der neu geschaffenen Benutzeroberfläche ist die Konfiguration an einem zentralen Punkt zusammengefasst. Die Konfiguration wurde abstrahiert und kann einfacher nachvollzogen und schneller bedient werden. Zusätzlich konnte die Navigation entschlackt werden. Die Anzahl Schaltflächen wurde reduziert, dies steigert nochmals den Bedienkomfort.

Das zweite Ergebnis zeigt sich nur im Hintergrund der Applikation. Um Mabata vom verwendeten Framework loszulösen wurde wie geplant eine Zwischenschicht eingebaut. Als Technologie für diese Middleware stellten sich WebServices als die beste Lösung heraus. Um die Kommunikationsschicht von Mabata loszulösen wurde diese durch einen Adapter ersetzt. Dieser Adapter nimmt die Methodenaufrufe von Mabata entgegen, wandelt sie in einen XML-String um und übergibt diese per WebService an die Middleware. Die Middleware erstellt aus dem erhaltenen XML-String wiederum Objekte und entscheidet, welche Aktionen auf diesen Objekten durchgeführt werden müssen. Dazu leitet die Middleware die Objekte an die entsprechende Managerklasse weiter. Diese Managerklassen übernehmen die Kommunikation mit dem Communications Manager. Hierfür wird das bestehende myFramework verwendet welches vorher aus Mabata herausgelöst wurde.

Das Services Framework, als weiteren Punkt auf der Aufgabenliste, konnte nur bis zum Design hin ausgearbeitet werden. Eine Umsetzung wurde nach den ersten beiden oben genannten Ergebnissen eingeplant. Dadurch, dass Mabata bei einem Kunden relativ früh installiert werden soll, wurde unser Zeitplan etwas zusammen gepresst. Um einen ersten Prototyp rechtzeitig fertig zu stellen, wurde die Umsetzung des Services Frameworks von unseren Betreuern übernommen. Die dadurch gewonnene Zeit konnte in intensive Tests und etlichen kleinere Core Verbesserungen investiert werden. Die wichtigsten Core Verbesserungen sind unter anderem, dass die Passwörter nicht mehr im Klartext in der Datenbank abgelegt werden. Weiter wurden einige grafische Unschönheiten ausgebügelt.

3.3. Schlussfolgerungen

Durch die erfolgreiche Umsetzung der Mabata Business Case Vereinfachung ist es einem Benutzer ohne grosse Vorkenntnisse möglich, Mabata zu bedienen. Dank der Loslösung der strikten Konfigurationsstruktur des Communications Managers ist die Bedienung intuitiver geworden. Dies ermöglicht die Applikation schneller einführen zu können. Die Endbenutzerschulung vereinfacht sich stark und die Akzeptanz steig.

Durch die Zentralisierung der Konfiguration werden mehr Daten pro Sicht geladen. Dies kann, sofern viele Daten vorhanden sind, zu Verzögerungen führen. Dies war vorher nicht der Fall, da die Konfiguration in viele kleine Schritte unterteilt war. Darum wurden immer nur wenige Datensätze benötigt. Die Anzahl Klicks für eine erfolgreiche Konfiguration lag aber deutlich höher. In einem späteren Schritt könnte dieses Problem untersucht und die Performance optimiert werden. Als Ansatz könnten die verschiedenen Konfigurationsarten für das Laden der Daten aus der Datenbank genauer analysiert werden.

Die komplette Benutzeroberfläche musste für diese Umsetzung überarbeitet werden. Dies führte dazu, dass umfangreiche Tests nötig wurden um die gesamte Funktionalität zu testen. Durch diese intensiven Tests wurde die Stabilität enorm gesteigert.

Vor der Umsetzung der Middleware konnte noch nicht genau abgeschätzt werden, wie gross der Performanceverlust durch die zusätzliche Kommunikationsschicht sein wird. Nach den ersten Tests mit einer frühen Version zeigte sich aber, dass die Middleware nur geringe Verzögerungen verursacht, und dies zu keinem grossen Performanceverlust führte. Das Auslagern der Logik für die Verarbeitung der Mabata Business Cases in die Middleware musste zwangsläufig durchgeführt werden, damit der Punkt Mabata Business Case Vereinfachung umgesetzt werden konnte. Dies ermöglicht zudem, weitere Applikationen auf diese Logik aufzusetzen. Als Ausblick für die Middleware kann sicher die Anbindung weiterer Applikationen und Zielsystemen in Angriff genommen werden. Eine gute Grundlage wurde dafür in dieser Bachelorarbeit erstellt.

Durch das Wegfallen der Umsetzung des Services Frameworks konnte mehr Zeit in die grundlegende Stabilität von Mabata investiert werden. Somit ist Mabata jetzt an einem Punkt, der einen produktiven Einsatz ermöglicht.

Projektplan

Im nachfolgenden Teil der Dokumentation wird die Projektplanung behandelt, welche folgendermassen unterteilt ist:

- Projektstruktur
- Management Abläufe
- Risiko Management
- Arbeitspakete
- Funktionsbeschrieb

4. Projektplan

4.1. Projektstruktur

Das Projektteam besteht aus 3 Teammitgliedern. Auf eine hierarchische Struktur wird verzichtet. Es wird jeweils ein Verantwortlicher bestimmt, welcher für einen Bereich die Verantwortung übernimmt. Dies heisst allerdings nicht, dass er der Alleinige Entwickler für diesen Bereich ist.

4.1.1. Teammitglieder

- Fabio Looser, fl
  - Bereich: Middleware
- Pascal Fuchs, pf
  - Bereich: Benutzeroberfläche
- Ueli Kuratli, uk
  - Bereich: Mabata Adapter

4.1.2. Betreuer

- Beat Stettler, bs
- René Stanger, rs
- Maurin Egler, me

4.2. Management Abläufe

4.2.1. Zeitmanagement

Die Bachelorarbeit startet am 16. Februar 2009 und dauert bis zum 12. Juni 2009. Der Projektumfang beträgt total ungefähr 1110 Stunden. Dies entspricht für die ersten 14 Arbeitswochen jeweils 20 Stunden pro Teammitglied. Für die letzten beiden Wochen werden 45 Arbeitsstunden pro Teammitglied berechnet.

4.2.2. Zeitplan

Siehe «Projektplan.xls».

4.2.3. Projektablaufplan

Der Projektablauf wird im MS Project festgehalten. Die ersten 6 Wochen bestehen darin, alle Vorarbeiten für die Implementation durchzuführen, bestehend aus Analyse, Design und Requirement Spezifikation. Danach folgt der grösste Aspekt der Arbeit, die Implementationsphase. Am Anfang werden die beiden Tasks «New VoIP Source Architecture» und «Mabata Business Case Vereinfachung» parallel implementiert. Darauf folgen die restlichen Tasks. Zum Schluss werden die Abschlussarbeiten durchgeführt.

Für den detaillierten Projektablaufplan wird auf «Projektplan.mpp» verwiesen.

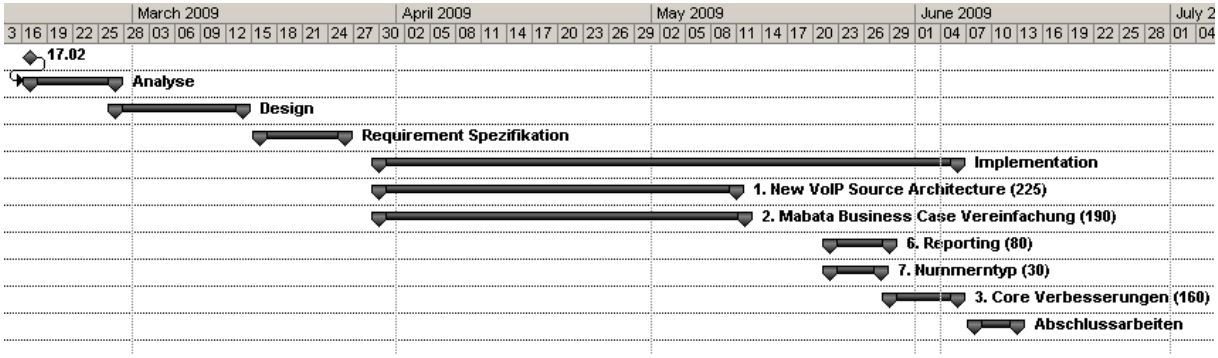

Abbildung 1 - Projektablauf

4.2.4. Iterationsplanung

Für die ersten Wochen, in welchen das Erstellen der Analyse und des Designs im Mittelpunkt steht, wird interativ vorgegangen. Dies bedeutet, dass jeweils eine Woche Zeit für eine Entwicklung einer Lösung bleibt, diese besprochen wird, und dann anhand der Ergänzungen erweitert wird.

Im späteren Verlauf des Projektes, falls eine neue Funktion implementiert werden soll, wird immer anhand der Komplexität dieser neuen Funktion entschieden, wie viel Zeit in die Analyse und das Design investiert wird.

4.2.5. Meilensteine

In den nachfolgenden Meilensteinbeschreibungen wird der Funktionsumfang nur stichwortartig beschrieben. Die einzelnen Funktionen der Meilensteine sind in der Anforderungsspezifikation detailliert beschrieben.

4.2.5.1. MS1 (Mittwoch, 25. März 2009)

- Installation der Infrastruktur
- Entwicklungsumgebung eingerichtet und lauffähig
- Projektplan mit Meilensteindefinitionen
- Review Analyse
- Review Anforderungsspezifikation
- Review Design

4.2.5.2. MS2 (Mittwoch, 22. April2009)

- Business Case Vereinfachung Administrator-Ansicht implementiert
- New VoIP Source Architecture für Administrator-Ansicht implementiert

4.2.5.3. MS3 (Donnerstag, 11. Mai 2009)

- Business Case Vereinfachung vollständig implementiert
- New VoIP Source Architecture vollständig implementiert

4.2.5.4. MS4 (Donnerstag, 1. Juni 2009)

- Reporting implementiert
- Nummerntyp implementiert
- SoftKey Template implementiert
- Displayname für Line implementiert
- Passwort-Hashes implementiert
- Erweiterung Nummerndatentyp implementiert

4.2.5.5. MS5 (Freitag, 12. Juni 2009)

- Testing/Bugfixing
- Schlussabgabe

4.2.6. Prototyp

Während der Bachelorarbeit soll ein Prototyp abgegeben werden, damit die aktuellen Fortschritte begutachtet und Feedback sowie Verbesserungsvorschläge eingebracht werden können.

Der Prototyp wird am 11.05.2009 mit folgenden Funktionen abgegeben:

- New VoIP Source Architecture implementiert
- Mabata Business Case Vereinfachung implementiert

4.2.7. Besprechungen

Die Projektsitzungen finden jeweils mittwochs um 14.00 Uhr mit dem ganzen Team und den Betreuern statt. Weitere Sitzungen des Teams können, falls nötig, spontan abgehalten werden. Das Team arbeitet montags bis donnerstags an seinem Arbeitsplatz im Raum 1.258. Hier kann sich das Team jeweils absprechen.

4.3. Risiko Management

ID	Risiko	Auswirkung	Massnahme	Wahrscheinlichkeit	Priorität
R01	Datenverlust	Schon Erledigte Arbeiten müssen nochmals gemacht werden	Keine Daten lokal speichern, sondern alles auf dem SVN- Server	niedrig	mittel
R02	Know How auf einzelne Personen konzentriert	Bei Ausfall einer Person muss sich ein anders Teammitglied in den Bereich einarbeiten	Häufige Kommunikation zwischen den Teammitgliedern	niedrig	mittel
R03	Ausfall der Infrastruktur	Weiterentwicklung wird verzögert	Absprache mit INS Mitarbeiter	mittel	hoch
R04	Ausfall eines Teammitglieds	Know How Verlust, Zeitplan kann nicht eingehalten werden	Knowledge Transfer im Team	niedrig	mittel
R05	Probleme mit externen Libraries und Software	Zeitverlust / geplante Umsetzung nicht möglich	Reservezeit einplanen	hoch	mittel
R06	Fehlerhafte Zeitplanung	Zuwenig Zeit um alle Funktionen zu Implementieren	Zeitreserven einplanen	mittel	mittel
R07	Interne Teamarbeit funktioniert nicht	Zeitverluste und Qualität des Projektes	Häufige Kommunikation im Team und Meinungsverschiedenheiten frühzeitig ausdiskutieren	niedrig	hoch
R08	Zu grosser Performanceverlust bei Übertragung der Daten über eine zusätzliche Zwischenschicht	Applikation reagiert sehr langsam. Im täglichen Gebrauch nicht mehr verwendbar.	Frühzeitiges Implementieren der WebService Schnittstelle	mittel	mittel

Tabelle 4 Risiko Management

4.4. Eingetroffene Risiken

4.4.1. R03

Während der Arbeit trat mehrmals das Phänomen auf, dass Grundfunktionen aus Mabata, welche bis zu dem Zeitpunkt problemlos funktionierten, nicht mehr ordnungsgemäss ausgeführt werden konnten. Nach dem die AXL-Nachrichten zwischen Communications Manager und Mabata analysiert wurden, zeigte sich, dass auf dem Communications Manager keine neuen Telefone mehr erstellt werden konnten. Es stellte sich heraus, dass Lizenzprobleme auf dem Server aufgetreten sind und dieser deshalb zurückgesetzt werden musste. Nach dem Rücksetzen musste dieser jeweils wieder neu konfiguriert werden.

Da uns dieses Problem bereits aus der Studienarbeit bekannt war, resultierten daraus keine nennenswerten Verzögerungen.

4.4.2. R06

Während dem Meilenstein 3 wurde uns von den Projektbetreuern mitgeteilt, dass sie den Prototyp aus dem Meilenstein 3 frühestmöglich wünschen. Aus diesem Grund wurde beschlossen, den Projektplan anzupassen und die Osterferien durchzuarbeiten. Dadurch konnte der Prototyp eine Woche vorher fertiggestellt werden.

4.5. Arbeitspakete

4.5.1. Projekt Management

Das Projekt Management beinhaltet den Aufbau und Betrieb der benötigten Infrastruktur sowie die gesamte Projektplanung.

4.5.2. Einarbeitung

Dieses Arbeitspaket umfasst die Einarbeitung in neue Themengebiete und Technologien.

4.5.3. Analyse

Neue Funktionen werden analysiert und mögliche Lösungen konzeptuell ausgearbeitet.

4.5.4. Design

In diesem Arbeitspaket werden die neuen Funktionen ausgearbeitet.

4.5.5. Requirements

Die Anforderungen an die neuen Funktionen werden definiert und schriftlich festgehalten.

4.5.6. Implementation

Dieses Arbeitspaket ist in die verschiedenen Funktionen unterteilt und enthält die Implementation der jeweiligen Funktionen.

4.5.7. Testing/Bugfixing

Tests sind Bestandteil dieses Arbeitspaketes. Hinzu kommt die Fehlerbehebung. Die Testprotokolle befinden sich im Verzeichnis «Dokumentation/Testprotokolle».

4.5.8. Abschlussarbeiten

Die Abschlussarbeiten beinhalten die Fertigstellung aller Dokumente und Tätigkeiten.

4.5.9. Dokumentationen

Für den Betrieb und Installation nötige Dokumente werden erstellt.

4.5.10. Sitzungen

Um die Arbeit innerhalb des Teams und zwischen den Betreuern abzugleichen.

4.6. Aufgabenstellung

Die Bachelorarbeit setzt sich aus drei verschiedenen Phasen zusammen. Die erste Phase stellt die Analyse dar, darauf folgen die Design- und Implementationsphase.

4.6.1. Analyse

In der Analyse befassen wir uns mit verschiedenen Problemstellungen, welche unten genauer erläutert werden.

4.6.1.1. New VoIP Source Architecture

Das INS hat mehrere eigene Softwareprodukte welche verschiedene Frameworks nutzen. Wenn sich in einem Framework Änderungen ergeben, müssen alle Applikationen geändert werden, welche das Framework benutzen.

Um dies zu verhindern, soll eine neue Zwischenschicht entwickelt werden, in welcher die Änderungen eines Frameworks zentral angepasst werden können. Zu beachten gilt, dass die Applikationen und Frameworks in verschiedenen Programmiersprachen geschrieben wurden.

4.6.1.2. Undo-Funktion

Alle in Mabata durchgeführten Aktionen sollen protokolliert werden. Zusätzlich können einzelne oder mehrere Änderungen rückgängig gemacht und somit ein früherer Systemzustand wiederhergestellt werden.

4.6.1.3. Services Framework

Das bestehende Services Framework soll ebenfalls über die neu erstellte Zwischenschicht angebunden werden und eigenständig lauffähig betrieben werden können.

4.6.1.4. Mabata Business Case Vereinfachung

Die bestehende Mabata Oberfläche ist zu wenig von der Konfiguration des Communications Managers abstrahiert. Es soll die Verwaltung des Communications Managers vereinfacht werden ohne Funktionalität einzubüssen.

4.6.1.5. Weitere Funktionen

Für folgende weitere Funktionen wird eine Analyse erstellt:

- Kostenabrechnung
- Reporting / Syslog
- Nummerntyp
- PhoneButton- und SoftKey Templates
- Modularisierung
- Displayname für Line
- Passwörter als Hash-Wert ablegen
- Erweiterung des Nummerndatentyps

4.6.2. Design

Für die oben beschriebenen Probleme wird aus der Analyse ein Designvorschlag erarbeitet, welcher in einer späteren Phase umgesetzt werden könnte.

4.6.3. Implementation

Da die Zeit für die Implementierung nicht für alle Funktionen reicht, mussten diese nach Prioritäten eingeordnet werden, und danach bestimmt werden, welche dieser Funktionen nun implementiert werden. Dies wird im folgenden Kapitel beschrieben.

4.7. Funktionsumfang

Aufgrund der noch zur Verfügung stehenden Zeit können nicht alle Funktionen implementiert werden. Um zu analysieren welche Funktionen prioritär umgesetzt werden müssen und wie viel Zeit dafür beansprucht wird, wurde eine detailierte Taskliste ausgearbeitet. Diese ist im Anschluss an dieses Kapitel ersichtlich.

4.7.1. Verfügbare Zeit:

Die für Implementationen noch zur Verfügung stehende Zeit wurde in der Tabelle abgebildet. Daraus ergibt sich ein Total von 588 Stunden welche noch auf die Funktionen aufgeteilt werden können.

Tätigkeit	Anzahl Wochen	Zu Stunden	Studenten	Total Stunden
Implementation	8	20	3	480
Implementation	1	45	3	135
Sitzung	9	1	3	-27
			Total	**588**

Tabelle 5 Verfügbare Zeit

4.7.2. Kurzübersicht Tasks

Die einzelnen Tasks spiegeln die Funktionen wieder, welche im Kapitel 5 Design im Dokument «Technische Referenz» ausgearbeitet wurden.

Task	Total Stunden	Prio
New VoIP Source Architecture (Middleware)	225	1
Mabata Business Case Vereinfachung	190	1
Services Framework mit Mabata	160	2
Services Framework ohne Mabata	60	3
Modularisierung	280	4
Kostenabrechnung	100	4
Reporting	80	1
Nummerntyp	30	1
Nummerndatentyp	30	2
Core Verbesserungen	130	2
Total:	**1125**	

Tabelle 6 Kurzübersicht Tasks

4.7.3. Definitiver Funktionsumfang

Es wurde beschlossen, dass alle Prio 1 sowie die Prio 2 Funktionen in dieser Bachelorarbeit umgesetzt werden. Das Services Framework wurde von unseren Betreuern übernommen.

4.8. Tätigkeitsliste

	Tasks			Beschreibung	Stunden
1.	**New VoIP Source Architecture**			Die neue Middleware, welche zwischen Mabata und den Communications Manager gesetzt wird. ***Abhängigkeiten: Zusammen mit Task 2***	**Tot.: 225**
	1.1	Methodenliste Mabata		Alle Methodenaufrufe, welche über die Kommunikationsschnittstelle abgewickelt werden, müssen erfasst und während der Entwicklung berücksichtigt werden.	5
	1.2	WebService			
		1.2.1	WebService erstellen	Pro Funktion wird ein WebService zur Verfügung gestellt, welcher Methodenaufrufe entgegennimmt und an die Business Logik weiterleitet.	10
		1.2.2	Authentifizierung	Mittels Hashfunktion eine Authentifizierung bereitstellen	5
	1.3	Business Logik		Verarbeitet die eingegangenen Daten	
		1.3.1	CRUD Mandant	Business Logik für die Behandlung eines Mandanten Aufrufes	
			Add/Get/Update/ Delete		je 5
		1.3.2	CRUD User	Business Logik für die Behandlung eines User Aufrufes	
			Add		20
			Get/Update/Delete		je 10
	1.4	myFramework Anbindung		Loslösung des myFramework aus Mabata und Integration in die Middleware	10
	1.5	Konfiguration (ohne Konfigurationstool)		Erarbeitung einer zentralen Konfigurationsstelle für alle benötigten Konfigurationsparameter.	10
	1.6	Mabata Adapter		Quellsysteme werden über sogenannte Adapter an die Middleware angebunden. Für das bestehende Quellsystem Mabata muss ein Adapter entwickelt werden, damit es mit der neuen Middleware kommunizieren kann.	
		1.6.1	Anbindung Middleware	Kommunikation zwischen dem Adapter und den WebServices der Middleware herstellen	10
		1.6.2	Objekt zu XML	Generierung von einem XML-String aus den Objekten, damit dieser über das Netzwerk übertragen werden kann	20
		1.6.3	XML zu Objekt	Parsing des erhaltenen XML-Strings und Erstellung eines Objektes daraus	20
		1.6.4	Konfiguration	Integration der Konfiguration des Adapters in den bestehenden Konfigurationsassistenten von Mabata.	5
		1.6.5	Integration in Mabata	Schnittstellenanpassung an Mabata: die Kommunikationsschicht, welche vorher aus dem myFramework bestand wird herausgelöst und durch den neuen Adapter ersetzt.	10
	1.7	Transaction Handling		Muss an Sitzung genau definiert werden.	30
	1.8	Reserve		10 % Reservezeit	20

2.	**Mabata Business Case Vereinfachung**			Die bestehende Mabata Oberfläche ist zu wenig von der Konfiguration des Communications Managers abstrahiert. Mit diesem Task soll die Verwaltung des Communications Managers vereinfacht werden ohne Funktionalität einzubüssen. ***Abhängigkeiten: Zusammen mit Task 2***	**Tot.: 190**
	2.1	GUI		Das GUI muss stark verändert werden. Diverse Schritte der Konfiguration werden unter einer gemeinsamen Ansicht zusammengefasst.	
		2.1.1	Mandant	Der Mandant kann neu an einem zentralen Ort konfiguriert werden.	30
		2.1.2	User	Der User kann neu an einem zentralen Ort konfiguriert werden.	40
	2.2	Controlleranpassungen		Auch im Core von Mabata werden Änderungen nötig, um mehrere Transaktionen zusammenzufassen und als neue Business Case Transaktion an die Middleware weiterzuleiten. (Daher auch die Abhängigkeit zu Task 1)	
		2.2.1	Mandant	Controlleranpassungen für den Mandant	
			Add/Get/Update/ Delete		je 5
		2.2.2	User	Controlleranpassungen für den User	
			Add		10
			Get/Update/ Delete		je 5
	2.3	Transaction Handling		Aktuelles Transaction Handling erweitern, da neu mehrere Objekte gleichzeitig auf den Communications Manager geschrieben werden	30
	2.4	SoftKey Template			
		2.4.1	GUI Anpassungen	Für die SoftKey Templates besteht noch keine Möglichkeit, diese im GUI zu konfigurieren.	5
		2.4.2	myFramework Erweiterung	Momentan gibt es im myFramework noch keine Funktion, um alle SoftKey Templates aus dem Communications Manager auszulesen.	5
		2.4.3	Erstellung Installationsanleitung	Die SoftKey Templates müssen direkt auf dem Communications Manager erstellt werden, dafür wird eine detailierte Installationsanleitung erstellt.	5
	2.5	PhoneButton Template mit Busy Lamp			
		2.5.1	GUI Anpassungen	Das GUI muss dahingehend geändert werden, dass nur noch die kompatiblen PhoneButton Templates zur Auswahl angezeigt werden.	5
		2.5.2	Erstellung Installationsanleitung	Die für Busy Lamp benötigten PhoneButton Templates müssen direkt auf dem Communications Manager erstellt werden. Dafür wird eine detailierte Installationsanleitung erstellt.	5
	2.6	Reserve		10 % Reservezeit	20

3.	Services Framework			*Abhängigkeiten: Task 1 muss umgesetzt werden*	**Total: 220 Mit Mabata: 160 Ohne Mabata: 60**
	3.1	Reduzierung bestehendes Services Framework		Die alte Version des Services Framework war in Mabata integriert, dies muss jetzt entfernt werden.	15
	3.2	Datenbank		Das Services Framework benötigt eine eigene Datenbank um die Services zu speichern	
		3.2.1	Datenbank erstellen	Erstellung des Datenbankschemas und der Datenbank selbst	10
		3.2.2	Anbindung	Die Datenbank wird mit dem Services Framework verbunden	10
	3.3	Konfiguration		Zentrale Lösung für die Konfiguration der benötigten Schnittstellen (Datenbank, Middleware-WebService, Mabata WebService)	10
	3.4	GUI		Erstellung eines neuen GUIs für die einzelnen Benutzerrollen	
		3.4.1	Admin		20
		3.4.2	Mandant		15
		3.4.3	User		10
	3.5	Anbindung GUI in Mabata		Das GUI wird von Mabata verwendet um die Services zu konfigurieren.	10
	3.6	Anbindung Mabata DB via WebService		Die Mabata DB wird benötigt, um die Mandanten sowie die Beziehung zwischen User und Mandant abzufragen.	
		3.6.1	WebService implementieren	Damit das Services Framework nicht direkt auf die Mabata Datenbank zugreift, werden die benötigten Daten über einen WebService bereitgestellt.	10
		3.6.2	Methodenauflistung	Alle Aufrufe benötigten eine WebService Methode. Damit keine Methode vergessen geht, werden alle Methoden gesammelt und während der Implementation berücksichtigt.	5
	3.7	Anbindung Middleware		Um Daten vom Communications Manager abrufen zu können, wird über die Middleware darauf zugegriffen.	20
	3.8	Anmeldung			10
	3.9	Services Framework ohne Mabata		Damit das Services Framework auch dann eingesetzt werden kann, wenn kein Mabata verwendet wird.	
		3.9.1	Datenbank Anpassungen		10
		3.9.2	GUI Anpassungen		20
		3.9.3	Userverwaltung		15
		3.9.4	Anmeldung		10
		3.9.5	Konfigurationsanpassungen		10
	3.10	Reserve		10 % Reservezeit	20

4.	Modularisierung			*Abhängigkeiten: Task 1 und 2 müssen umgesetzt werden*	Tot.: 280
	4.1	Erstellung Externe Applikation		Um Module installieren zu können, wird eine externe Applikation benötigt.	
		4.1.1	XML-Schemadefinition	Die Module werden in einer definierten XML-Datei erstellt. Diese Datei wird dann verwendet, um das GUI generisch zu erstellen.	20
		4.1.2	XSLT-Transformatoren	XSLT-Dateien werden benötigt, um das erstellte Schema nach vordefinierten Regeln in andere Zieldateien transformieren zu können.	20
		4.1.3	XSLT-Processing	Für die Transformation werden XSLT-Processoren benötigt, diese erstellen aus XML-Dateien mittels den XSLT-Dateien die benötigten Zieldateien.	10
		4.1.4	JSPX-Generierung	Eine Zieldatei ist das GUI des Modules, welches in JSPX-Dateien gespeichert ist.	30
		4.1.5	Java-Controller Generierung	Auch werden Java-Dateien benötigt, welche die Logik des neuen Modules beinhalten.	20
		4.1.6	Kompilation	Die erstellten Java-Dateien müssen kompiliert werden, damit diese verwendet werden können.	10
		4.1.7	Konfigurationsassistent anpassen	Die Konfiguration muss exportiert und importiert werden können	5
	4.2	Integration Modul			
		4.2.1	Generisches GUI Mabata	Das bestehende GUI von Mabata muss dahingehend angepasst werden, dass es generisch um weitere Module erweitert werden kann.	15
		4.2.2	Observer für Module	Die Controller der neuen Module müssen sich beim Haupt-Controller anmelden, damit der weiss, dass zusätzliche Module installiert wurden.	15
		4.2.3	Datenbank Erweiterung	In der Mabata Datenbank werden 2 neue Tabellen benötigt. Diese beinhalten alle Module, sowie die Zuweisung der Mandanten zu den Modulen	10
		4.2.4	Konfigurations-GUI	Der Administrator benötigt ein GUI, um die Module den Mandanten zuweisen zu können.	15
	4.3	Installation Modul			
		4.3.1	Installationsroutinen erstellen	Für die Installation eines zusätzlichen Modules wird eine Installationsroutine benötigt, welche die oben aufgeführten Tasks nacheinander abarbeitet.	15
		4.3.2	Datenbankeintrag erstellen	Die Tabelle welche alle Module beinhaltet, muss um den Eintrag des neu installierten Modules erweitert werden.	10
		4.3.3	Handling WAR-Datei	Die Installation der Module wird direkt in der WAR-Datei vorgenommen.	15
		4.3.4	Testing	Die Installation der Module ist ein kritischer Task, dieser Task muss ausführlich getestet werden.	20
	4.4	Anpassungen bestehendes GUI			
		4.4.1	Erstellung Modul Communications	Das bestehende GUI beinhaltet alles für die Konfiguration des Communications Managers.	20

			Manager	Dieses GUI muss in ein Modul ausgelagert werden.	
	4.5	Reserve		10 % Reservezeit	30
5.	**Kostenabrechnung**				**Tot.: 100**
	5.1	Datenbank Erweiterung		Es wird eine zusätzliche Tabelle benötigt.	10
	5.2	GUI Erweiterung			
		5.2.1	Admin	Der Administrator kann im GUI die Tagessätze sowie die Einrichtungspauschalen festlegen. Eine Übersicht über alle Mandanten und deren Kostenabrechnung ist zusätzlich möglich.	25
		5.2.2	Superuser	Der Superuser kann seine Kostenabrechnung ebenfalls online anschauen.	10
	5.3	Core Erweiterung		Jede Aktion die Kosten zu Folge hat, oder ändert, muss in der neuen Tabelle rapportiert werden.	15
	5.4	Export Funktion		Die Kostenabrechnung kann mittels CSV-Dateien exportiert werden.	10
	5.5	Externe Schnittstelle		Eine externe Schnittstelle wird für den Zugriff auf die Kostenabrechnung zur Verfügung gestellt (z.B. für die Buchhaltung). Mithilfe dieser Schnittstelle können Daten als CSV-String abgerufen werden.	20
	5.6	Reserve		10 % Reservezeit	10
6.	**Reporting**				**Tot.: 80**
	6.1	GUI Erweiterungen			
		6.1.1	Admin	Kann alle geloggten Daten betrachten sowie Einstellungen vornehmen	20
		6.1.2	Superuser	Kann seine geloggten Daten betrachten	10
	6.2	Anbindung Syslog-Server		Die Log-Daten sollen von einem Syslog-Server geloggt werden können.	15
	6.3	Core Erweiterungen		Alle Aktionen müssen geloggt werden können, je nach dem ob der Debug-Modus eingeschalten ist.	25
	6.4	Reserve		10 % Reservezeit	10
7.	**Nummerntyp**				**Tot.: 30**
	7.1	Anpassung GUI		Das GUI zeigt neu auch den Status der Nummer korrekt an, wenn diese einer Hunt- oder Pickup Gruppe gehört	15
	7.2	Anpassungen Datenbank		Damit der Status angezeigt werden kann, muss dies in der Datenbank berücksichtigt werden.	10
	7.3	Reserve		10 % Reservezeit	5
8.	**Nummerndatentyp**				**Tot.: 30**
	8.1	Ändern des Datentyps		Es sollen Nummern die mit 00 oder + beginnen unterstütz werden.	30
9.	**Core Verbesserungen**				**Tot.: 130**
	9.1	Passwort als Hash		Das Passwort soll als Hash-Wert in der Datenbank abgelegt werden.	20
	9.2	Importassistent erweitern		Der Importassistent soll für Hunt Gruppen erweitert werden. Zudem sollen die Nummern per SQL abgeholt werden.	40
	9.3	Unterstützung Softphones		Es müsen SoftPhones unterstützt werden.	20
	9.4	Diverse GUI Verbesserungen		Offensichtliche Unschönheiten im GUI werden verbessert.	50

Projektmonitoring

Im nachfolgenden Teil der Dokumentation wird das Projektmonitoring behandelt, welches folgendermassen unterteilt ist:

- Zeiterfassung
- Journal
- Sitzungsprotokolle
- Auswertung des Zeitplans
- Persönliche Berichte

5. Projekt Monitoring

5.1. Zeiterfassung

Die Zeiterfassung wurde in einem separaten Excel-Dokument unter folgendem Pfad geführt:

„Projektplan/Zeitplan.xls“

5.2. Journal

5.2.1. Meilenstein 1

Da der Start der Studienarbeit sehr schleppend vor sich ging, legten wir von Anfang an grossen Wert darauf, dass uns das gleiche Missgeschick nicht noch einmal passiert. Noch vor der ersten Sitzung, in welcher die Aufgaben der Bachelorarbeit genau definiert wurden, richteten wir die komplette Entwicklungsumgebung ein. In der ersten Sitzung wurde der grobe Ablauf der Arbeit besprochen sowie die ersten Teilprojekte, welche es auf die nächste Woche zu analysieren gab. Die ersten sechs Wochen liefen so ab, dass jeweils auf die Sitzung eine Aufgabe ausgearbeitet wurde, um diese dann den Betreuern vorzustellen. Danach entwickelte sich eine Diskussion über die vorgestellte Lösung. Diese Anmerkungen wurden in der darauffolgenden Woche in die bestehende Lösung eingearbeitet.

Dieser erste Projektabschnitt verlief aus unserer Seite gesehen sehr produktiv. Als äusserst interessant stellten sich die Sitzungen heraus, da häufig noch weiterführende Anforderungen an das Produkt gestellt wurden, welche noch einzuarbeiten waren.

Nach Abschluss dieses Meilensteins war eine sehr gute Grundlage für die Implementierungsphase geschaffen worden, da sehr viele Probleme schon vor ihrem Auftreten analysiert werden konnten.

5.2.2. Meilenstein 2

Nachdem die etwas länger dauernde erste Phase des Projektes abgeschlossen wurde, waren wir froh, endlich mit der Umsetzung der ausgearbeiteten Arbeitspaketen zu beginnen. Es tat gut, wieder einmal nicht mit Word, sondern mit Eclipse arbeiten zu können.

Eingeplant war, dass wir in diesem Meilenstein das GUI für die Mandator Oberfläche vereinfachen, sowie die Middleware für die Funktionen der Mandator Oberfläche fertigstellen.

Die für diesen Meilenstein benötigten neuen Technologien waren zum einen die WebService Technologie und zum anderen das Parsen der Objekte zu XML und wieder zurück. Da wir während der ersten Phase diese Technologien schon ein wenig untersucht hatten, wussten wir wie diese einzusetzen waren und stiessen auf keine all zu grossen Probleme.

Wir kamen gut voran und konnten schon bald eine erste funktionierende Version der Middleware fertigstellen. Danach begannen wir mit der Umstellung von Mabata auf die Middleware, um das gleichzeitig erarbeitete GUI für die Mandantenkonfiguration auch über die neue Middleware testen zu können. Diese Umstellung erwies sich als schwierig, da nicht einfach einzelne Teile umgestellt werden konnten, denn diese zogen oft weitreichende Änderungen mit sich. Es gelang uns trotzdem, die Umstellung erfolgreich über die Bühne zu bringen und den Meilenstein 2 erfolgreich abzuschliessen.

5.2.3. Meilenstein 3

Da der Meilenstein 2 fristgerecht abgeschlossen wurde, konnten die Arbeiten des dritten Meilensteins rechtzeitig in Angriff genommen werden. Kurz nach dem Start von diesem Meilenstein musste der Projektplan angepasst werden. Die Projektbetreuer wünschten, dass der Prototyp möglichst früh fertiggestellt wird, damit genügend Zeit für ein ausgiebiges Testen übrig bleibt. Aus diesem Grund haben wir uns entschlossen, die Osterferien durchzuarbeiten und die Abgabe dieses Meilensteins um eine Woche vorzuverlegen. Um das Funktionieren dieses Prototyps sicherzustellen wurde ein Testprotokoll erstellt und durchgearbeitet. Durch das Testprotokoll konnte ein funktionsfähiger und stabiler Prototyp abgegeben werden.

5.2.4. Meilenstein 4

In diesem Meilenstein wurden sehr viele Funktionen zusammengefügt, welche nicht viel Zeit kosteten aber durchaus einen Mehrwert erzielen. Ein paar dieser Funktionen tauchten erst im Laufe des Projektes auf. Einziges grösseres Teilprojekt in diesem Meilenstein war die Implementierung des Reportings. Dies bestand auf der einen Seite aus dem Design, wie die ganze Protokollierung ablaufen soll, und andererseits aus der etwas mühsameren Fleissarbeit, um alle Reporting-Einträge an der richtigen Stelle im Code anzubringen.

Während diesem Meilenstein wurden alle störenden GUI-Elemente oder Verhalten von Mabata, welche immer aufgeschoben worden sind, behoben. Damit konnte für den Endbenutzer von Mabata eine sehr wertvolle Verbesserung erzielt werden.

5.2.5. Meilenstein 5

Die Bachelorarbeit geht in die letzte heisse Phase. Da der reguläre Unterricht zu Ende ist, konnten wir uns voll und ganz auf die Bachelorarbeit konzentrieren. Die Implementierung aller Funktionen wurde bereits abgeschlossen. Nun konzentrierten wir uns auf das Testen und die Fehlerbehebung. Von unseren Betreuern erhielten wir eine Testumgebung, welche bereits vollständig mit Testdaten konfiguriert war. Damit fanden wir diverse Fehler und konnten diese beheben. Auch haben die Betreuer selbst auf einer eigenen Testumgebung diverse Tests durchgeführt, um noch weitere Fehler zu finden die wir dann beheben konnten. Diese Zeit erwies sich als sehr produktiv für das Endprodukt, so konnte die Stabilität enorm gesteigert werden. Während der Studienarbeit hatten wir leider nicht die Möglichkeit, solche umfangreiche Tests durchzuführen.

Zum Schluss musste die erstellte Dokumentation fertig gestellt und korrigiert werden. Dieser Punkt ist wichtig für unsere Nachfolger, damit sich diese schnell in Mabata zu Recht finden.

Auch diverse administrative Tätigkeiten wie das Erstellen eines Posters oder die Broschüre, in der alle Arbeiten vorgestellt werden, mussten erledigt werden.

Zum Schluss sind wir froh, dass die strenge Zeit vorbei ist. Wir haben viel gelernt und konnten von unserem, durch das Studium erarbeitete Wissen profitieren.

5.3. Sitzungsprotokolle

Die Sitzungsprotokolle der Sitzungen sind im Verzeichnis «Sitzungsprotokolle» abgelegt.

5.4. Auswertungen des Zeitplans

5.4.1. Soll/Ist-Vergleich der Arbeitszeit pro Woche

Durchschnittlich haben wir pro Woche ein paar Stunden mehr gearbeitet als eingeplant. In der Woche 9 waren offizielle Frühlingsferien. Diese haben wir, wie bereits geschrieben, für die Implementierung genutzt. Der Anstieg in den letzten beiden Wochen resultiert daher, dass kein Unterricht mehr stattfand.

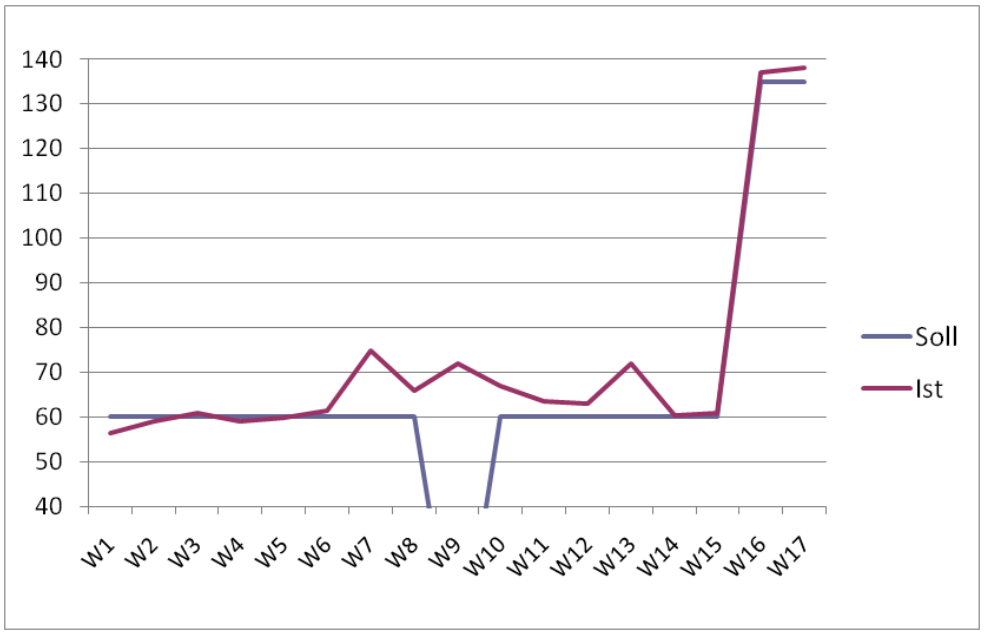

Abbildung 2 Soll/Ist Vergleich der Arbeitszeit pro Woche

5.4.2. Soll/Ist-Vergleich der Arbeitszeit pro Arbeitspaket

Die beiden Pakete Implementation sowie Testing/Bugfixing haben etwas mehr Zeit in Anspruch genommen wie geplant.

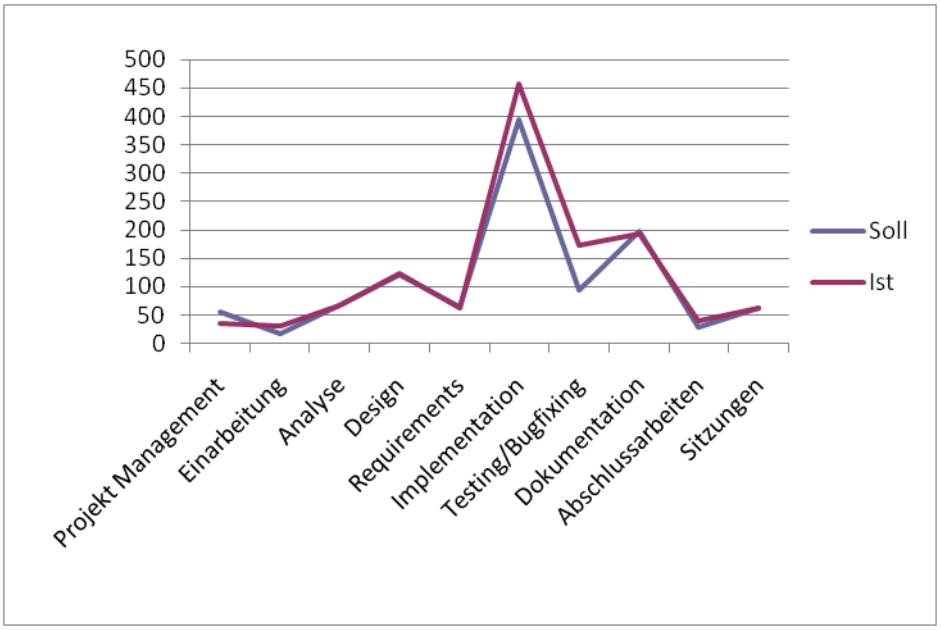

Abbildung 3 Soll/Ist-Vergleich der Arbeitszeit pro Arbeitspaket

5.4.3. Zeitaufwand pro Arbeitspaket

Diese Grafik zeigt den prozentualen Anteil der einzelnen Arbeitspakete.

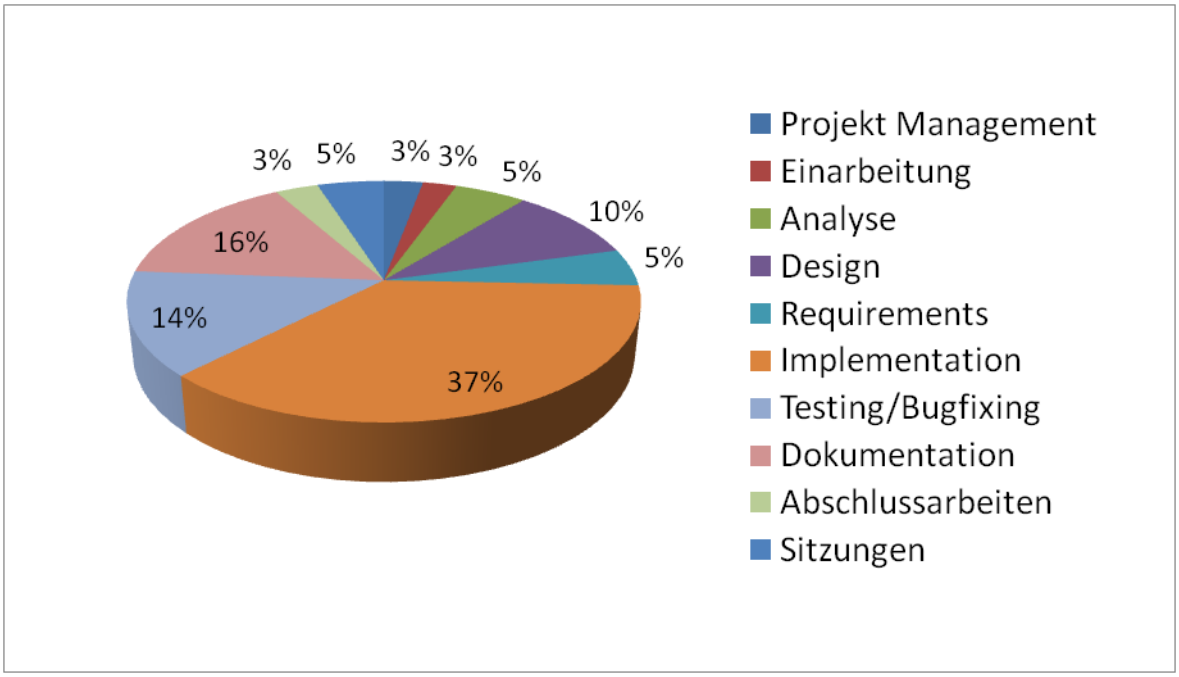

Abbildung 4 Zeitaufwand pro Arbeitspaket

5.5. Persönliche Berichte

5.5.1. Pascal Fuchs

5.5.1.1. Thema

Da es sich bei dieser Arbeit um die Fortsetzung der Studienarbeit handelte, war ich mit diesem Thema bereits von Anfang an vertraut. Bereits während der Studienarbeit fand ich die Arbeit an Mabata interessant und durch die Vielfalt der eingesetzten Technologien sehr abwechslungsreich. Aus diesen Gründen stand für mich schnell fest, dass Mabata das Thema meiner Bachelorarbeit sein wird.

Zudem erhielten wir durch diese Arbeit die Möglichkeit, Mabata für den ersten produktiven Einsatz fertigzustellen.

5.5.1.2. Projektverlauf

Der Projektstart verlief, anders als bei der letzten Arbeit, zügig und problemlos. Durch die Einführung der Middleware erstreckte sich die Analysephase über mehrere Wochen. Während dieser Analysephase waren die wöchentlichen Sitzungen mit den Projektbetreuern sehr hilfreich. Damit wir auf Entscheidungen jeweils nicht eine Woche warten mussten, hielten wir zum Teil auch zwei Sitzungen pro Woche ab. Dadurch konnten wir den strikten Zeitplan ohne grössere Probleme einhalten. Uns war bewusst, dass das ausgiebige Testen der Applikation sehr wichtig für den Einsatz von Mabata in einer produktiven Umgebung ist. Aus diesem Grund waren wir selbstverständlich bereit, auf unsere Frühlingsferien zu verzichten um dadurch den Prototyp eine Woche früher fertigzustellen als geplant.

5.5.1.3. Team

Durch die Tatsache, dass wir bereits mehrere Arbeiten in diesem Team erfolgreich durchgeführt hatten, erwartete ich auch für dieses Projekt eine gute Teamarbeit. Aus den Erfahrungen der Studienarbeit haben wir die Arbeiten nach groben Themenbereichen aufgeteilt. Durch diese Aufteilung konnte jedes Mitglied zum Teil grössere eigenständige Arbeiten ausführen. Dies war für uns auch sehr wichtig, da in diesem Semester die Stundenpläne der Teammitglieder unterschiedlich waren.

5.5.1.4. Fazit

Selbstverständlich haben unsere Zeitschätzungen aus der Projektplanung nicht immer zugetroffen, aber durch die Einplanung von genügend Reserve-Zeit konnten die Termine der Meilensteine trotzdem eingehalten werden.

Alles in allem ist diese Bachelorarbeit für mich ein voller Erfolg. Ich konnte viel Neues dazulernen und es entstand ein Produkt, welches aus meiner Sicht ein grosses Erfolgspotential aufweist. Zudem haben wir mit der Entwicklung der Middleware eine Grundlage für weitere interessante und sinnvolle Arbeiten geschaffen.

5.5.2. Ueli Kuratli

5.5.2.1. Thema

Durch die durchgeführte Studienarbeit war das Thema schon zum vornherein bekannt. Ich wusste daher was auf mich zukommt. Dank der neuen Middleware blieb es nicht nur bei der reinen Erweiterung vom bestehenden Mabata, sondern es kam auch ein völlig neuer Punkt hinzu. Diesen Punkt durften wir selbständig ausarbeiten und schlussendlich auch umsetzen. Dadurch kamen auch neue Technologien ins Spiel, zum einen haben wir intensiv mit WebServices gearbeitet, zum anderen konnte ich das JAXB-Framework anwenden.

Die Motivation nochmals eine Arbeit über Mabata zu erarbeiten war dadurch eigentlich schon gegeben. Ein weiterer Punkt war, dass nun Mabata endlich produktiv eingesetzt wird. Dadurch stieg die Motivation und wir hatten auch mehr Zeit, die Stabilität zu verbessern.

5.5.2.2. Projektverlauf

Den etwas holprigen Start in die Studienarbeit wollten wir nicht noch einmal wiederholen. So gingen wir von Anfang an motiviert an die Arbeit, um auch zu zeigen, dass wir es anders können. Die erste Phase war relativ lange, so vergingen die ersten sechs Wochen bis wir das erste Mal programmieren konnten. Dies war für mich eine strenge Zeit, da wir viel Design dokumentieren mussten.

Als wir mit dem Programmieren loslegten, ging es zügig vorwärts und wir konnten dank den Frühlingsferien, die wir auch zum Programmieren nutzten, den Prototyp relativ früh abgeben. Auch die uns zur Seite gestandenen Betreuer halfen uns vor allem im Bereich Testing enorm.

Zum Schluss konnten wir ohne allzu grossen Zeitdruck das Projekt fertig stellen. Ich bin sicher, dass es jetzt um einiges stabiler läuft als am Ende der Studienarbeit.

5.5.2.3. Team

Im Team konnten wir effizient arbeiten, da wir schon während der Studienarbeit und auch schon früher zusammen gearbeitet hatten. Jeder erhielt ein Spezialgebiet, für welches er verantwortlich war. Dadurch hatte jeder seinen Teilbereich, an dem er hauptsächlich arbeiten konnte. Da wir nun nicht mehr den gleichen Stundenplan hatten, kam es vor, dass wir nicht immer alle zusammen an der Bachelorarbeit arbeiteten. Durch eine erhöhte Kommunikation konnten wir auch dieses Problem lösen.

5.5.2.4. Fazit

Mit dem Ergebnis dieser Bachelorarbeit bin ich sehr zufrieden. Wir haben Mabata nun endlich soweit weiterentwickelt und stabilisiert, dass es in einem produktiven Umfeld eingesetzt werden kann. Durch die neue Middleware bleibt zudem genügend Raum für weitere Projekte offen.

In diesem Projekt hatten wir zum ersten Mal eine grössere Umstellung des Projektplans. Da wir aber flexibel genug waren, konnten wir die Zeit entsprechend um planen, damit wir den neuen Anforderungen des Zeitplanes gerecht wurden.

5.5.3. Fabio Looser

5.5.3.1. Thema

Da ich schon die Studienarbeit über dieses Thema gemacht habe, war für mich klar, auf was ich mich eingelassen habe. Das schöne an Mabata ist, das sehr viele verschiedene Technologien zusammentreffen, und von daher sehr lehrreich und abwechslungsreich ist.

Der grösste Motivationspunkt war, dass wir Mabata soweit entwickeln sollten, dass nun nach mehreren Arbeiten die Software das erste Mal produktiv eingesetzt werden sollte. Dabei konnte nicht einfach, wie teilweise im Code anzutreffen war, ein Hack eingebaut werden, um eine schöne Demo präsentieren zu können. Dies wäre mit Sicherheit schon von den Betreuern während der Testphase aufgedeckt worden.

5.5.3.2. Projektverlauf

Da der Start in die Studienarbeit sehr schlecht war, war ich für die Bachelorarbeit umso motivierter, zu zeigen, dass wir es auch anders können. Meiner Meinung nach ist uns dies sehr gut gelungen. In den ersten Wochen konnte eine sehr gute Grundlage aufgebaute werden, von welcher wir in der Implementierungsphase profitieren konnten. Meinem Empfinden nach waren die Projektsitzungen einiges konstruktiver wie während der Studienarbeit, da sehr viele verschiedene Aspekte diskutiert wurden, welche danach weiter ausgearbeitet werden konnten.

Als positiv empfand ich die Zusammenarbeit mit den Betreuern gegen den Schluss der Projektarbeit, in welcher in enger Zusammenarbeit das Testing in Angriff genommen wurde.

5.5.3.3. Team

Da ich mit diesem Team schon die Studienarbeit zusammen gemacht habe, war es für mich keine Überraschung, dass wir auch als Team gut funktionierten. Im Vergleich zur Studienarbeit haben wir die Zuständigkeiten stärker abgegrenzt. Somit war jeder auf einem Gebiet ein Experte. Dies funktionierte ausgezeichnet.

5.5.3.4. Fazit

Mit dieser Bachelorarbeit bin ich sehr zufrieden. Ich habe das Gefühl, dass wir einen grossen Beitrag dazu leisten konnten, damit Mabata produktiv bei Kunden installiert werden kann. Die aktuelle Mabata Version macht mir einen sehr kundenfreundlichen Eindruck und auch die Stabilität, welche nicht immer über alle Zweifel erhalten war, konnte sehr stark verbessert werden.

Als negativer Punkt ist die Zeitschätzung zu erwähnen, in welcher wir teilweise Funktionen zu tief einschätzen, obwohl wir eher das Gefühl hatten, zu viel Zeit eingeplant zu haben. Dieser Mehraufwand, welchen wir dadurch zu leisten hatten, war aber nicht wirklich schlimm, da wir das Glück hatten, ein Projekt bearbeiten zu dürfen, welches die grösste Zeit Freude bereitete.