

# MCM Multi Channel Mobile

## Dokumentation und Entwicklung einer Multi Channel Applikation mittels Java und UML



Gruppensemesterarbeit

Zürcher Fachhochschule

**HWZ Hochschule für Wirtschaft Zürich**

eingereicht bei:

Prof. Dr. Walter Kuhn & Stefan Berger

vorgelegt von: Pius Fonseca, Dieter Good, Marcel Styger

Studiengruppe: BWI-A11

Adressen: Dieter Good (Sportweg 7, CH-8704 Herrliberg)  
Pius Fonseca (Bodenacker 12, CH-8046 Zürich)  
Marcel Styger (Chüngengass 3, CH-8805 Richterswil)

Sourcen: <https://github.com/mstyger/MultiChannelMobile>

Google Play: <https://play.google.com/store/apps/details?id=ch.zh.dipima.multichannelmobile>

Zürich, den 15.01.2013

## Inhaltsverzeichnis

1. Einleitung .....	4
1.1 Ausgangslage.....	4
1.2 Aufgabenbeschreibung.....	4
1.3 Abgrenzung .....	4
2. Projektkonzept .....	5
2.1 Projektplan.....	5
2.1.1 Kommentar .....	5
2.2 Eingesetzte Mittel .....	6
2.3 Aufgabenpakete.....	6
3. Use Case Diagramm.....	7
4. Use Cases .....	7
4.1 SMS senden (UC1).....	7
4.1.1 Sequenzdiagramm .....	9
4.1.2 Aktivitätsdiagramm .....	10
4.2 Nachricht senden (UC2) .....	10
4.3 Nachricht validieren (UC3) .....	11
4.4 Log ansehen (UC4).....	11
4.4.1 Sequenzdiagramm .....	12
4.5 Info ansehen (UC5).....	12
4.5.1 Sequenzdiagramm .....	13
5. Klassendiagramm.....	14
5.1 Diagramm.....	14
5.2 Erläuterungen.....	15
6. Android Activity Life Cycle.....	16
7. Test-Cases .....	17
8. Featureliste .....	18
9. Installationsanleitung Android.....	19
10. Benutzer Manual.....	19
10.1 App herunterladen / Programmstart .....	19
10.2 Email senden.....	20
10.3 SMS senden / MMS senden .....	21
10.4 Nachricht zum Ausdrucken .....	23

10.5	Nachrichtenlog anzeigen .....	24
10.6	Info anzeigen .....	24
11.	Schlusswort .....	25
11.1	Fazit .....	25
11.2	Lernbericht je Teilnehmer .....	26
11.2.1	Pius Fonseca .....	26
11.2.2	Dieter Good .....	26
11.2.3	Marcel Styger .....	27
12.	Abbildungsverzeichnis .....	28

# 1. Einleitung

## 1.1 Ausgangslage

Diese Gruppensemesterarbeit ist im Herbstsemester 2012 / 2013 anhand eines Lernauftrages in den Modulen „Java 1 & 2“ und „Software Engineering“ an der Hochschule für Wirtschaft Zürich (HWZ) entstanden. Sinn und Zweck dieses Lernauftrages ist es, die erlernten Techniken der Java-Programmierung und UML-Modellierung in die Praxis umzusetzen.

## 1.2 Aufgabenbeschreibung

Konzeptionierung und Erstellung einer Software, die Nachrichten auf unterschiedlichen Arten versenden kann. Eine Nachricht soll als SMS, MMS, Email oder Print (Druck) versendet werden. Bei allen 4 Kanälen müssen Validierungen durchgeführt werden, damit der mögliche Transport gewährleistet werden kann. Neben den vorgegebenen Anforderungen waren mindestens zwei Extras selber zu definieren.

Als erstes Extra haben wir uns für die Implementierung einer Android-App entschieden. So kann die App direkt auf einem Android-Mobiltelefon verwendet und entwickelt werden. Als zweites Extra haben wir eine Logfunktion eingebaut. In diesem Log werden alle versendeten Nachrichten gespeichert und können per Knopfdruck wieder aufgerufen werden.

## 1.3 Abgrenzung

In dieser Arbeit beschränken wir uns lediglich auf die Themen Java und UML bzw. auf die für uns relevanten Klassen. Die zahlreichen Android Schnittstellen und Bibliotheken werden nur am Rande erwähnt und sind nicht im Fokus dieser Arbeit.

## 2. Projektkonzept

### 2.1 Projektplan

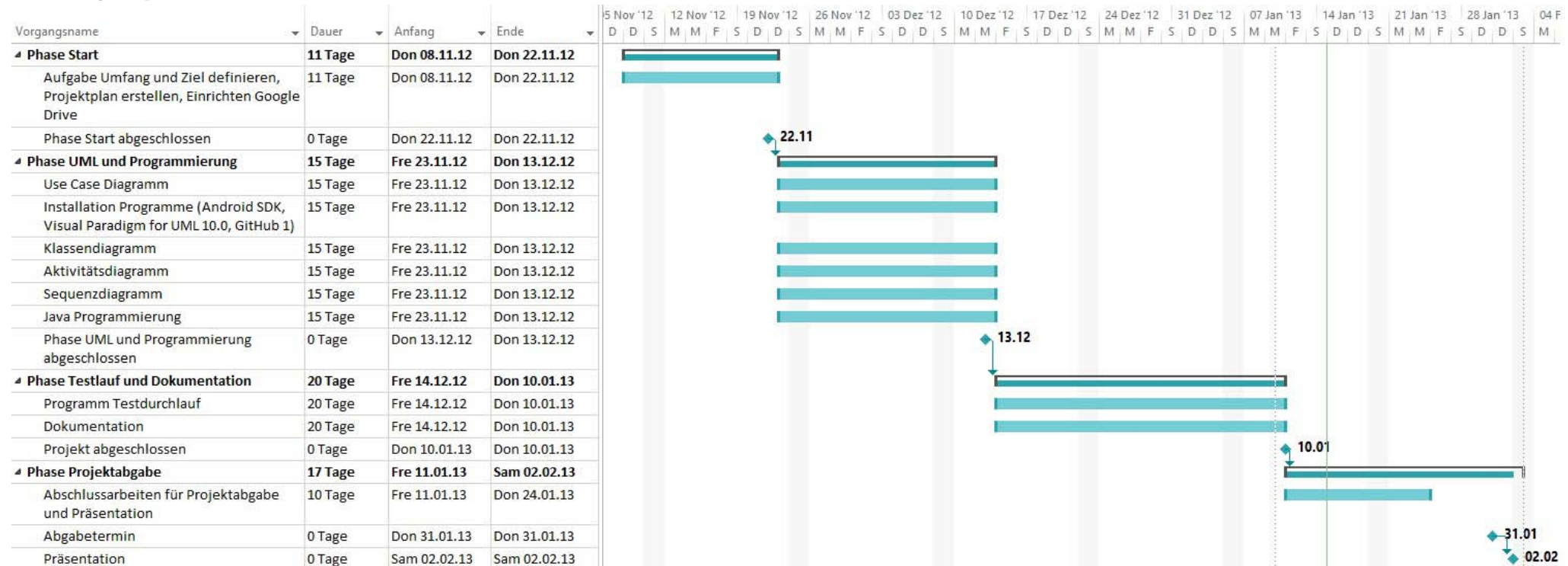


Abbildung 1: Projektplan

#### 2.1.1 Kommentar

Beim Erstellen des Projektplans wurden als erster Schritt die Termine für unsere Meilensteine festgelegt. Dann haben wir die Pflichtaufgaben gemäss Projektauftrag eingeplant. Den Projektplan konnten wir einhalten und in verschiedenen Fällen waren wir sogar früher fertig als geplant. Dies war zum Beispiel beim Programmieren der Android - App oder beim Fertigstellen unserer Dokumentation der Fall. Die effiziente Arbeitsweise ist wohl nicht zuletzt auch auf die gut aufgebauten Kommunikationswege innerhalb der Teammitglieder zurück zu führen und dem Know-how von Marcel Styger als Lead unserer Gruppe. Über GitHub, Skype bis hin zu Google Drive und WhatsApp waren viele technische Hilfsmittel im Einsatz.

## 2.2 Eingesetzte Mittel

In der untenstehenden Tabelle sind alle Hilfsmittel und Technologien aufgelistet, die im Verlauf des Projektes eingesetzt wurden. Mit den aufgezeigten Mitteln konnten wir uns, trotz räumlicher Trennung, bestens organisieren und immer den aktuellsten Stand des Projektes miteinander abgleichen.

Mittel	Beschreibung / Einsatz
<b>GitHub</b>	Versionenkontrolle. Gemeinsamen Code über zentralen Datenspeicher verwalten. ( <a href="https://github.com/mstyger/MultiChannelMobile">https://github.com/mstyger/MultiChannelMobile</a> )
<b>Eclipse (mit Android SDK)</b>	Java Entwicklungsumgebung mit Android Entwicklungserweiterung
<b>Google Drive</b>	Filesharing-Plattform, Ablage von allen Projektdokumenten
<b>MS Project 2013</b>	Erstellung und Pflege des Projektplanes
<b>MS Visio</b>	Darstellung von Diagrammen (Feinschliff)
<b>Skype</b>	Konferenzsitzungen über Internettelefonie. Regelmässig durchgeführte Sessions, um den Projektstatus abzugleichen (min. 1x pro Woche)
<b>WhatsApp</b>	Instantmessaging Dienst. Kurze News über Änderungen (z.B. Neu geschriebener Code, Skype-Session-Ankündigung, etc.)
<b>Visual – Paradigm Agilian</b>	UML Modellierung
<b>Object Aid</b>	Eclipse integriertes UML Werkzeug

## 2.3 Aufgabenpakete

Der Grossteil der Aufgaben wurde zusammen vorbesprochen und meist auch zusammen umgesetzt. Einzelne Aufgaben wurden untereinander aufgeteilt und anschliessend zusammengeführt. Nachfolgend eine Kurzübersicht der Haupttätigkeiten:

Marcel Styger	Dieter Good	Pius Fonseca
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> SW-Architektur	<input type="checkbox"/> SW-Implementierung	<input type="checkbox"/> SW-Implementierung
<input type="checkbox"/> SW-Implementierung	<input type="checkbox"/> Nachricht Validierung	<input type="checkbox"/> Nachricht Logfunktion
<input type="checkbox"/> Andorid-Integration	<input type="checkbox"/> Konzept / Doku	<input type="checkbox"/> Use Cases
<input type="checkbox"/> Klassendiagramm	<input type="checkbox"/> Aktivitätsdiagramm	<input type="checkbox"/> Aktivitätsdiagramm
<input type="checkbox"/> Sequenzdiagramm	<input type="checkbox"/> Sequenzdiagramm	<input type="checkbox"/> Sequenzdiagramm
<input type="checkbox"/> Aktivitätsdiagramm	<input type="checkbox"/> Testing	<input type="checkbox"/> Testing
<input type="checkbox"/> Testing		

### 3. Use Case Diagramm

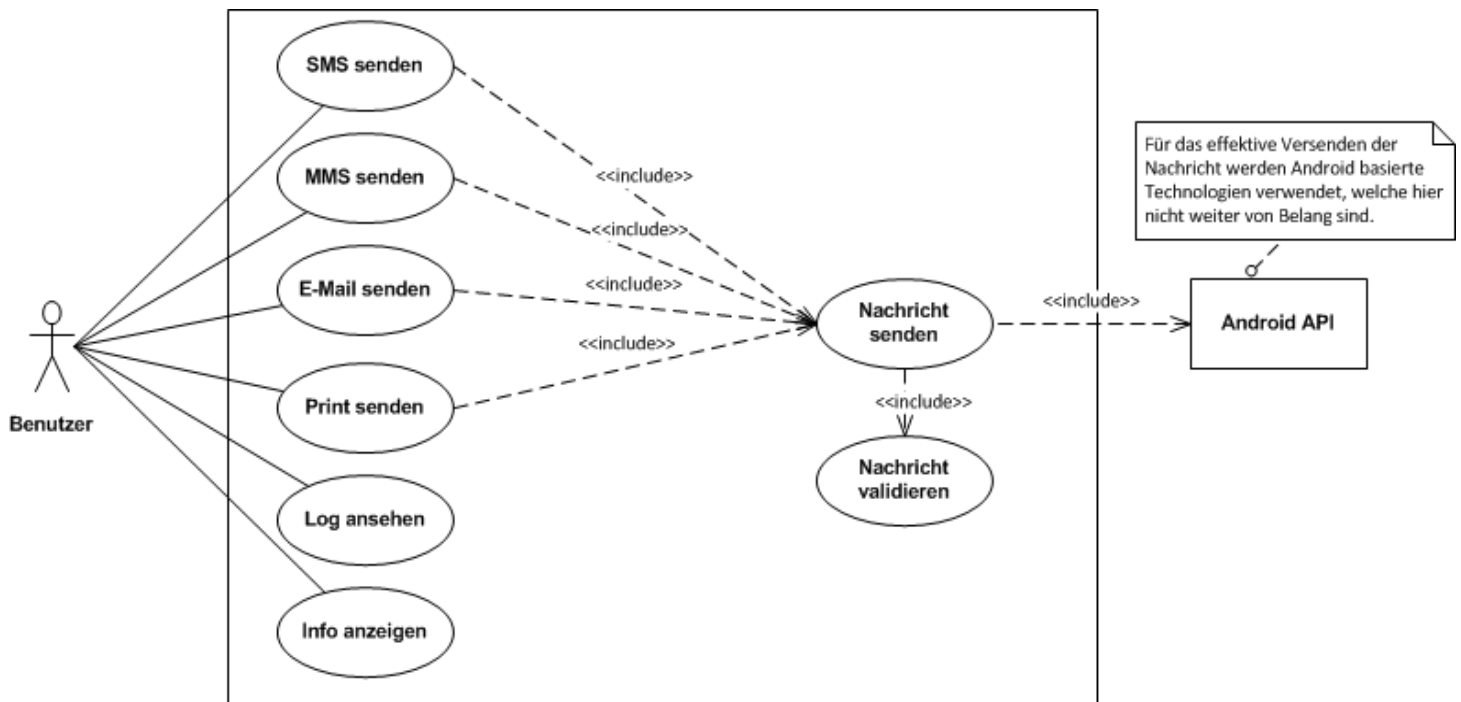


Abbildung 2: UC Diagramm

### 4. Use Cases

#### 4.1 SMS senden (UC1)

Dieser Use Case wird stellvertretend für das Senden aller Nachrichtstypen (MMS, E-Mail, Print) verwendet.

<b>Use Case ID:</b>	UC1		
<b>Use Case Name:</b>	SMS senden		
<b>Created By:</b>	Dipima	<b>Last Updated By:</b>	Dipima
<b>Date Created:</b>	12.12.2012	<b>Last Revision Date:</b>	20.12.2012
<b>Actors:</b>	Benutzer		
<b>Description:</b>	Der Benutzer wählt den Typ der Nachricht, welchen er versenden möchte. Nach Wahl des Nachrichtstyps werden entsprechende Felder für das Erfassen der Nachricht angezeigt.		
<b>Trigger:</b>	Benutzer tippt in der Home-Activity <sup>1</sup> auf den Button „SMS senden“.		
<b>Preconditions:</b>	Das App wurde fehlerlos gestartet und die Buttons für das Wählen des Nachrichtstyps werden angezeigt.		
<b>Postconditions:</b>	Die Felder der Nachricht wurden durch den Benutzer wunschgemäß ausgefüllt und sind bereit für das Versenden.		
<b>Normal Flow:</b>	Benutzer tippt auf „SMS senden“ in der Home-Activity, gibt den Empfänger und den Text ein in der dabei geöffneten Formularansicht und tippt auf „SMS senden“.		
<b>Alternative Flows:</b>	Androidspezifische Interaktionen, wie ein eingehender Telefonanruf		

<sup>1</sup> Startaktivität der App

	und der Gleichen, werden hier nicht spezifisch behandelt. Dies regelt das Betriebssystem selber.
<b>Exceptions:</b>	-
<b>Includes:</b>	Use Case "Nachricht senden"
<b>Frequency of Use:</b>	Unbeschränkt.
<b>Special Requirements:</b>	Android Version 3.1 sollte auf dem ausführenden Gerät mindestens vorhanden sein, damit eine optimale Usability erreicht werden kann.
<b>Assumptions:</b>	-
<b>Notes and Issues:</b>	-



#### 4.1.1 Sequenzdiagramm

Im folgenden Sequenzdiagramm werden die Objekte visualisiert, die im Zuge des Versendens einer Nachricht generiert werden. Dieses Sequenzdiagramm zeigt den ganzen Prozess des Versendens einer Nachricht, vom wählen des Nachrichtentyps bis zur Validierung der Nachricht.

In der MainActivity wird die App gestartet und über die onCreate – Methode initialisiert. In der WriteMessage – Klasse wird das GUI und die Message - Klassen initialisiert.

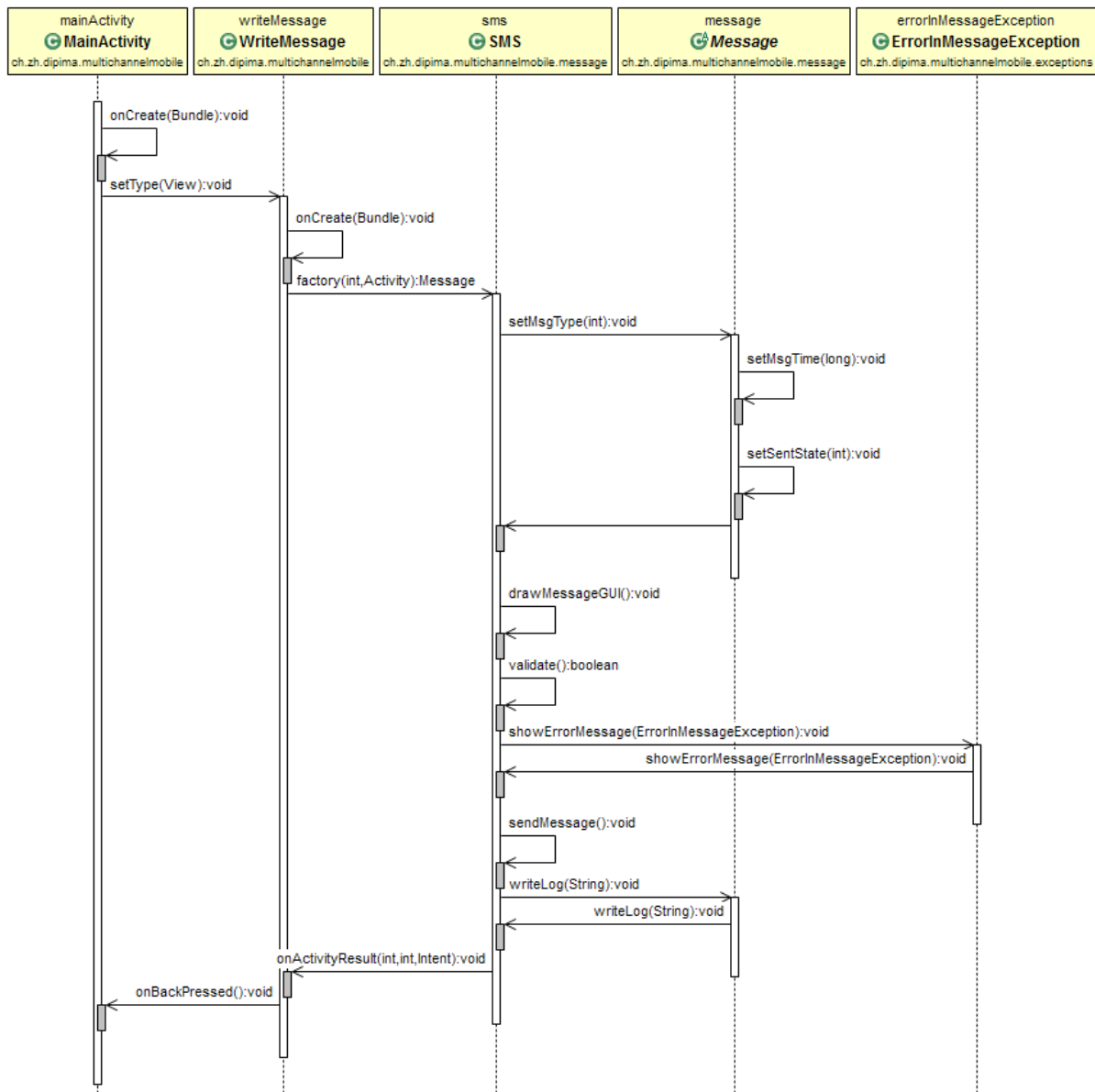


Abbildung 2: Sequenzdiagramm Nachricht senden

#### 4.1.2 Aktivitätsdiagramm

Das Aktivitätsdiagramm zeigt die Verantwortlichkeit einzelner Prozesse beim Versenden einer Nachricht sowie deren Reihenfolge und Abhängigkeit.

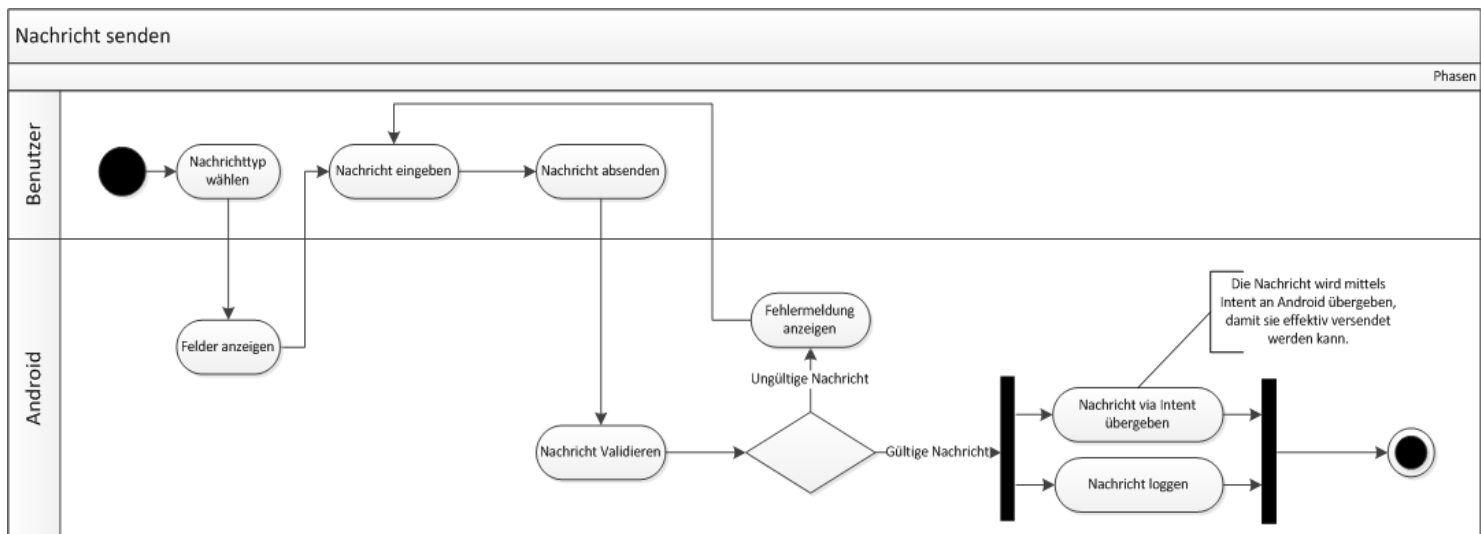


Abbildung 3: Aktivitätsdiagramm Nachricht senden

#### 4.2 Nachricht senden (UC2)

<b>Use Case ID:</b>	UC2		
<b>Use Case Name:</b>	Nachricht senden		
<b>Created By:</b>	Dipima	<b>Last Updated By:</b>	Dipima
<b>Date Created:</b>	12.12.2012	<b>Last Revision Date:</b>	20.12.2012
<b>Actors:</b>	Benutzer (indirekt) <sup>2</sup>		
<b>Description:</b>	Die Nachricht wird bei allen Nachrichtstypen nach betätigen des Senden-Buttons validiert, an Android zur weiteren Versendung weitergegeben und geloggt.		
<b>Trigger:</b>	Benutzer klickt in der Formularansicht auf senden.		
<b>Preconditions:</b>	Der Benutzer hat einen Nachrichtstyp gewählt und die Felder wunschgemäß abgefüllt.		
<b>Postconditions:</b>	Die Validierung wird ausgewertet (UC3 wird inkludiert).		
<b>Normal Flow:</b>	Der Benutzer füllt alle Felder wunschgemäß aus und betätigt den Senden-Button, worauf die Nachricht validiert und bei erfolgreicher Validierung dann versendet wird. Am Schluss wird die Nachricht geloggt.		
<b>Alternative Flows:</b>	Die Validierung ergibt einen Fehler. In diesem Fall wird eine Meldung ausgegeben und die Nachricht wird nicht versendet bzw. an Android zur weiteren Verwertung weitergereicht.		
<b>Exceptions:</b>	Diese werden in Use Case Validierung behandelt. Im Fehlerfall wird das Senden abgebrochen.		
<b>Includes:</b>	Nachricht validieren		

<sup>2</sup> Benutzer klickt den Vorgang zwar an, der Prozess wird jedoch von Android bzw. UC1 ausgeführt. Ohne Benutzer keine Interaktion, daher „indirekt“.

<b>Frequency of Use:</b>	-
<b>Special Requirements:</b>	Permissions für das Senden von Nachrichten in Android. Dies muss im Android Manifest deklariert werden und der Benutzer muss diese Aktivität erlauben.
<b>Assumptions:</b>	-
<b>Notes and Issues:</b>	-

#### 4.3 Nachricht validieren (UC3)

<b>Use Case ID:</b>	UC3		
<b>Use Case Name:</b>	Nachricht validieren		
<b>Created By:</b>	Dipima	<b>Last Updated By:</b>	Dipima
<b>Date Created:</b>	12.12.2012	<b>Last Revision Date:</b>	20.12.2012
<b>Actors:</b>	Benutzer (indirekt)		
<b>Description:</b>	Vor dem Versenden einer Nachricht soll sie geprüft werden. Konkret wird geprüft, ob alle nötigen Felder korrekt ausgefüllt wurden.		
<b>Trigger:</b>	Dieser Use Case wird durch UC2 eingeleitet. Ursprünglicher Trigger ist der Benutzer, welcher auf den Senden-Button klickt.		
<b>Preconditions:</b>	Siehe UC2.		
<b>Postconditions:</b>	-		
<b>Normal Flow:</b>	UC2 initiiert diesen Use Case, welcher die Felder nach Fehlern überprüft.		
<b>Alternative Flows:</b>	-		
<b>Exceptions:</b>	Ein Feld erfüllt nicht die gewünschten Inhalte. Dies führt zu einem Fehler, welcher den Sendeprozess terminiert.		
<b>Includes:</b>	-		
<b>Frequency of Use:</b>	-		

#### 4.4 Log ansehen (UC4)

<b>Use Case ID:</b>	UC4		
<b>Use Case Name:</b>	Log ansehen		
<b>Created By:</b>	Dipima	<b>Last Updated By:</b>	Dipima
<b>Date Created:</b>	12.12.2012	<b>Last Revision Date:</b>	20.12.2012
<b>Actors:</b>	Benutzer		
<b>Description:</b>	Die versendeten Nachrichten, welche geloggt werden, können eingesehen werden.		
<b>Trigger:</b>	Der Benutzer tippt in der Home Activity auf „Log anzeigen“		
<b>Preconditions:</b>	Siehe UC1		
<b>Postconditions:</b>	-		
<b>Normal Flow:</b>	Benutzer tippt in der Home-Activity auf „Log anzeigen“		
<b>Alternative Flows:</b>	-		
<b>Exceptions:</b>	-		
<b>Includes:</b>	-		
<b>Frequency of Use:</b>	-		

#### 4.4.1 Sequenzdiagramm

Dieses Sequenzdiagramm veranschaulicht das Anzeigen des Nachrichtenlogs. Die displayLog – Methode liest das Logfile und übergibt es mittels Adapter dem Frontend. Durch den onBackPressed – Aufruf wird der Vorgang abgeschlossen und terminiert.

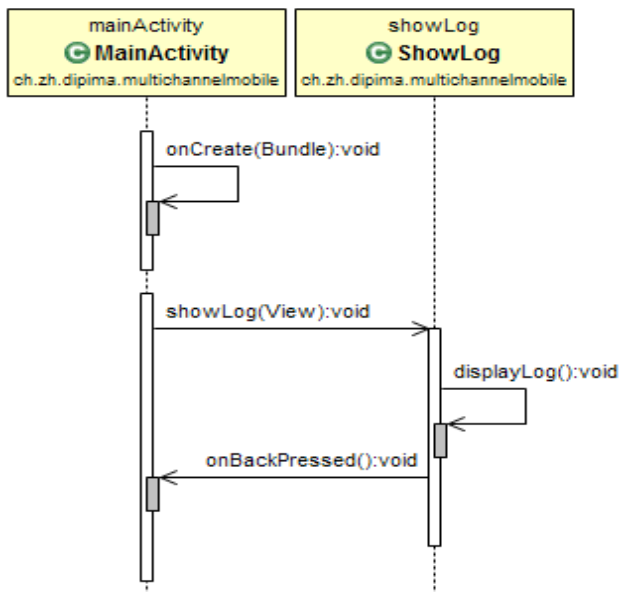


Abbildung 4: Sequenzdiagramm Log anzeigen

#### 4.5 Info ansehen (UC5)

<b>Use Case ID:</b>	UC5		
<b>Use Case Name:</b>	Info anzeigen		
<b>Created By:</b>	Dipima	<b>Last Updated By:</b>	Dipima
<b>Date Created:</b>	12.12.2012	<b>Last Revision Date:</b>	20.12.2012
<b>Actors:</b>	Benutzer		
<b>Description:</b>	Der Benutzer kann sich Informationen zum App anzeigen lassen.		
<b>Trigger:</b>	Der Benutzer tippt auf das Info-Icon in der Titlebar der Home Activity		
<b>Preconditions:</b>	Siehe UC1		
<b>Postconditions:</b>	-		
<b>Normal Flow:</b>	Der Benutzer tippt auf das Info-Icon in der Titlebar der Home Activity		
<b>Alternative Flows:</b>	-		
<b>Exceptions:</b>	-		
<b>Includes:</b>	-		
<b>Frequency of Use:</b>	-		

#### 4.5.1 Sequenzdiagramm

In diesem Sequenzdiagramm wird das Anzeigen der App – Info visualisiert. Mittels Intent wird beim Klicken in einem Menü die Info-Activity aufgerufen, die sich in der onCreate – Methode initialisiert und das Layout – XML verarbeitet.



Abbildung 5: Sequenzdiagramm Info anzeigen

## 5. Klassendiagramm

### 5.1 Diagramm

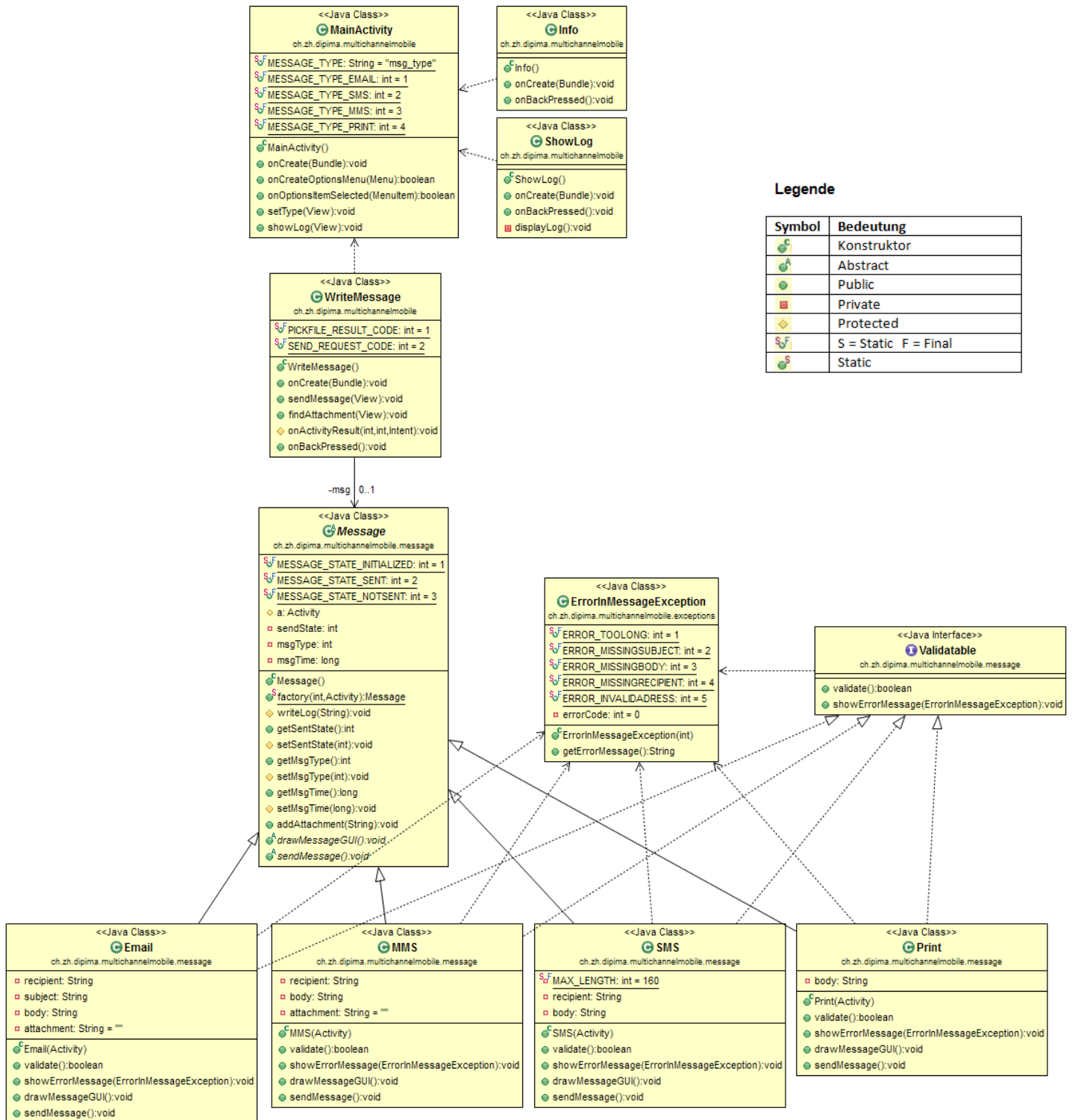


Abbildung 6: Klassendiagramm

## 5.2 Erläuterungen

Das Smartphone initialisiert die App mit der MainActivity. Dort wird mittels XML das GUI aufgebaut. Der Kern der Logik dieser Anwendung ist in der Klasse Message sowie ihrer Unterklassen zu finden. Die abstrakte Klasse Message beinhaltet die Activity, die für Android spezifische Operationen verwendet wird und von Subklassen überschrieben werden kann sowie private Felder, die für alle Nachrichtentypen geltenden Eigenschaften beinhalten (sendState, msgType, msgTime).

Die typischen Eigenschaften der einzelnen Nachrichtentypen werden dann in den von Message abgeleiteten Klassen definiert. Alle diese Subklassen implementieren das Interface Validatable, welches das Validieren vor dem Senden einer Nachricht verlangt. Zwecks Fehlermeldung ist die Exception ErrorInMessageException erstellt worden, welche von den einzelnen Nachrichtentypen geworfen werden kann.

Der Klassenmember, der den Text der Nachricht beinhaltet, wurde bewusst nicht in die Mutterklasse aufgenommen, da alle Nachrichtentypen eigene Zeichensätze haben könnten und dies nicht verallgemeinert werden darf.

Alle Nachricht - Objekte sollen über die statische Factory erstellt werden, diese generiert die entsprechenden Objekte anhand der in der MainActivity definierten Nachrichtentypen.

## 6. Android Activity Life Cycle

Für das Entwickeln von Android Applikationen ist das Verständnis des Activity Life Cycles von zentraler Bedeutung. Es zeigt den Lebenszyklus einer Activity von der Erstellung bis zur „Zerstörung“ (onDestroy()).

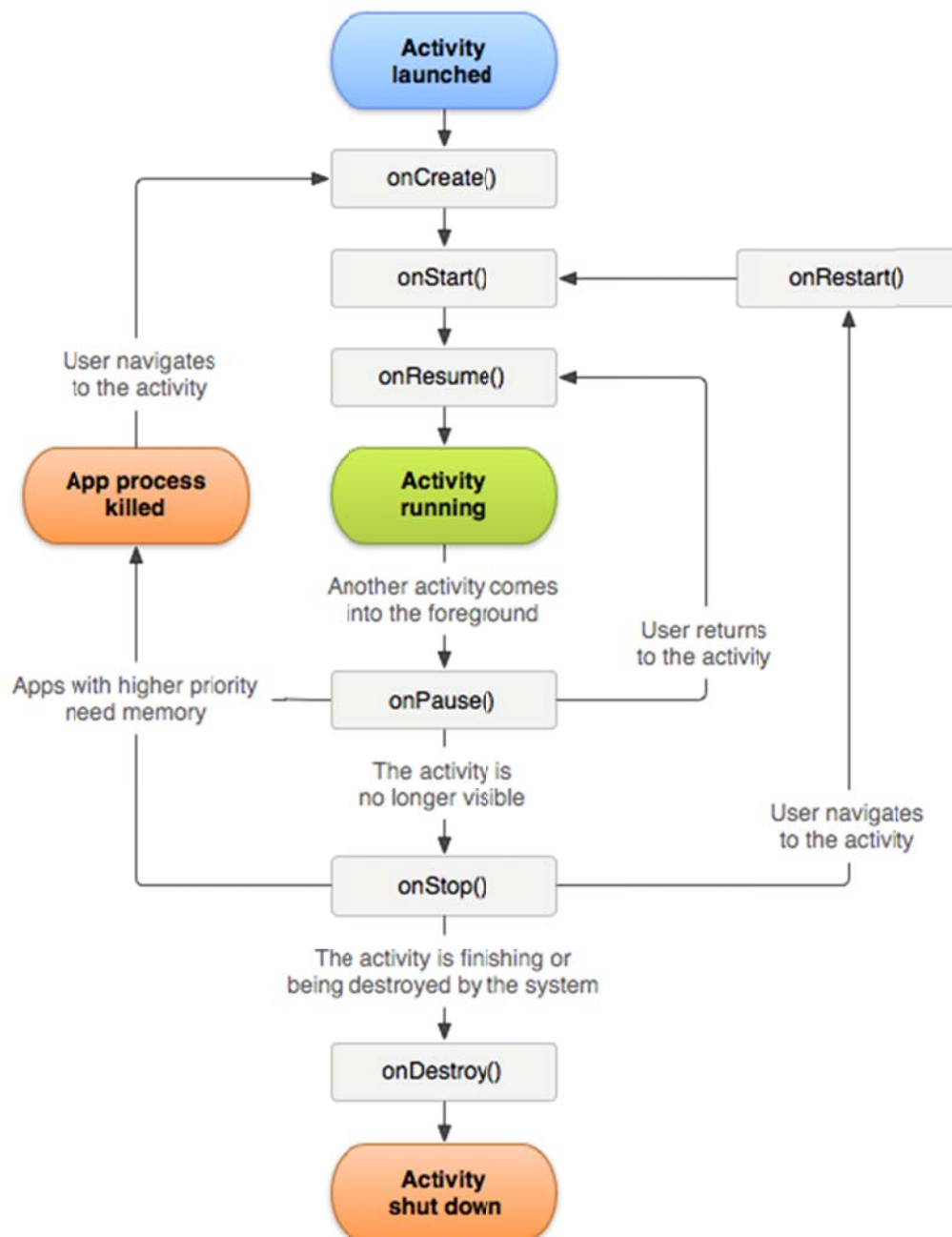


Abbildung 7: Android Activity Life Cycle, Quelle: <http://developer.android.com>



## 7. Test-Cases

Test-ID	Titel	Beschreibung	Trigger	Erwarteter Output	Effektiver Output	Status
1	App starten	Die App wird auf dem Google Samsung Nexus (Android 4.1.1) gestartet ohne Fehler	Tab auf das App-Icon	App startet	App startet	
2	Email vorbereiten	Beim betätigen des Buttons "Email senden" wird eine neue Activity gestartet, bei welcher ein Email verfasst werden kann.	Tab auf den Button "Email senden" in der Startactivity	Activity startet	Activity startet	
3	Email validieren	Folgende Felder werden geprüft, und eine Fehlermeldung wird ausgegeben, wenn sie leer sind: Empfänger, Betreff, Nachricht. Bei einer ungültigen Empfängeradresse wird ebenfalls ein Fehler ausgegeben.	Felder ausfüllen und "Email senden" betätigen	Validierung startet (Fehler werden ausgegeben im Fehlerfall oder Message wird an Android übergeben, wenn erfolgreiche Validierung)	Validierung startet (Fehler werden ausgegeben im Fehlerfall oder Message wird an Android übergeben, wenn erfolgreiche Validierung)	
4	Email senden	Email wird an Android weitergegeben, damit es von einem Email-Client gesendet werden kann.	Wenn Validierung erfolgreich, wird automatisch an Android weitergegeben	Benutzer kann Email-Client von Android zum senden der Nachricht wählen und Log wird geschrieben	Benutzer kann Email-Client von Android zum senden der Nachricht wählen und Log wird geschrieben	
5	SMS vorbereiten	Beim betätigen des Buttons "SMS senden" wird eine neue Activity gestartet, bei welcher ein SMS verfasst werden kann.	Tab auf den Button "SMS senden" in der Startactivity	Activity startet	Activity startet	
6	SMS validieren	Folgende Felder werden geprüft, und eine Fehlermeldung wird ausgegeben, wenn sie leer sind: Empfänger, Nachricht.	Felder ausfüllen und "SMS senden" betätigen	Validierung startet (Fehler werden ausgegeben im Fehlerfall oder Message wird an Android übergeben, wenn erfolgreiche Validierung)	Validierung startet (Fehler werden ausgegeben im Fehlerfall oder Message wird an Android übergeben, wenn erfolgreiche Validierung)	
7	SMS senden	SMS wird an Android weitergegeben, damit es von einem SMS-Client gesendet werden kann.	Wenn Validierung erfolgreich, wird automatisch an Android weitergegeben (Benutzer kann Client selber wählen)	Benutzer kann SMS-Client von Android zum senden der Nachricht wählen und Log wird geschrieben	Benutzer kann SMS-Client von Android zum senden der Nachricht wählen und Log wird geschrieben	
8	MMS vorbereiten	Beim betätigen des Buttons "MMS senden" wird eine neue Activity gestartet, bei welcher ein MMS verfasst werden kann.	Tab auf den Button "MMS senden" in der Startactivity	Activity startet	Activity startet	
9	MMS validieren	Folgende Felder werden geprüft, und eine Fehlermeldung wird ausgegeben, wenn sie leer sind: Empfänger, Nachricht.	Felder ausfüllen und "MMS senden" betätigen	Validierung startet (Fehler werden ausgegeben im Fehlerfall oder Message wird an Android übergeben, wenn erfolgreiche Validierung)	Validierung startet (Fehler werden ausgegeben im Fehlerfall oder Message wird an Android übergeben, wenn erfolgreiche Validierung)	
10	MMS senden	MMS wird an Android weitergegeben, damit es von einem MMS-Client gesendet werden kann.	Wenn Validierung erfolgreich, wird automatisch an Android weitergegeben (Benutzer kann Client selber wählen)	Benutzer kann MMS-Client von Android zum senden der Nachricht wählen und Log wird geschrieben	Benutzer kann MMS-Client von Android zum senden der Nachricht wählen und Log wird geschrieben	
11	Druck vorbereiten	Beim betätigen des Buttons "Nachricht zum Ausdrucken" wird eine neue Activity gestartet, bei welcher ein Print verfasst werden kann.	Tab auf den Button "Nachricht zum Ausdrucken" in der Startactivity	Activity startet	Activity startet	
12	Druck validieren	Folgende Felder werden geprüft, und eine Fehlermeldung wird ausgegeben, wenn sie leer sind: Nachricht.	Felder ausfüllen und "Nachricht zum Ausdrucken" betätigen	Validierung startet (Fehler werden ausgegeben im Fehlerfall oder Message wird an Android übergeben, wenn erfolgreiche Validierung)	Validierung startet (Fehler werden ausgegeben im Fehlerfall oder Message wird an Android übergeben, wenn erfolgreiche Validierung)	
13	Druck senden	Es wird die Meldung ausgegeben, dass die Nachricht gedruckt wurde (es wird nicht effektiv gedruckt, in der Version 1.0 wird darauf verzichtet, weil es nicht effektiver Teil der Aufgabenstellung ist).	Wenn Validierung erfolgreich, wird automatisch ein Toast angezeigt	Meldung wird angezeigt, dass gedruckt wurde	Meldung wird angezeigt, dass gedruckt wurde	
14	Nachrichtenlog anzeigen	Es soll eine History über alle gesendeten Nachrichten angezeigt werden	Tab auf den Button "Nachrichtenlog anzeigen"	Log wird angezeigt	Log wird angezeigt	
15	App-Info anzeigen	Es sollen App-Metainformationen angezeigt werden	Tab auf das Info-Icon in der Titelbar	Infoactivity öffnet	Infoactivity öffnet	
16	Android Navigation	Generell soll über den Backbutton und den Homebutton die "androidüblichen" Navigationen getätigt werden können	Tab auf den Back- bzw. Homebutton	Androidtypische Navigationsverhalten (Activity-Historie wird zurück navigiert beim Tab auf den Back-Button und die App schliesst beim klicken auf den Home-Button)	Androidtypische Navigationsverhalten (Activity-Historie wird zurück navigiert beim Tab auf den Back-Button und die App schliesst beim klicken auf den Home-Button)	

	Keine Fehler
	Verbessern in nächster Generation
	Fehler

## 8. Featureliste


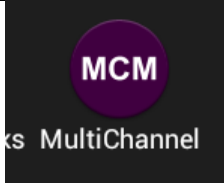
#	Beschreibung	Prio.	Realisiert	Kommentar
1	Android App entwickeln	2	Ja	
2	Nachrichttyp vor Eingabe der Nachricht auswählen (Im Home Activity)	1	Ja	
3	Nachrichttyp erst nach Eingabe der Nachricht auswählen	1	Nein	Zu umständliche Programmierung in Android (Überprüfen von Feldern, Ziffern und Sonderzeichen)
4	Empfängereingabe überprüfen	1	Ja	Wird bei jedem Typ auf entsprechendes Format überprüft (Email: @ und .xx Format, MMS / SMS: Nummerneingabe)
5	Überprüfung der Felder, ob Eingabe gemacht wurde	1	Ja	Überprüfung bei: Nachricht, Betreff (MMS / Email), Empfänger. Wenn ein Feld leer ist wird nichts gesendet und eine FM erscheint.
6	Textlänge überprüfen bei SMS	1	Ja	Bei max. 160 Zeichen erscheint eine Fehlermeldung
7	Automatische Umwandlung von SMS zu MMS wenn Text zu lang	3	Nein	In neuem Release. Zurzeit erscheint Warnung dass Text zu lang.
8	Nummern / Adressen aus Telefonbuch auslesen	3	Nein	In neuem Release
9	Anhänge anfügen bei MMS und Email	2	Zum Teil	Anhang kann ausgewählt werden, jedoch wird er noch nicht weiterversendet. Wird im neuen Release implementiert.
10	Gesendete Nachrichten via Log auslesen	3	Ja	Es werden alle Nachrichten von allen Nachrichtstypen in das gleiche File gespeichert und können so ausgelesen werden.
11	Nachrichten können versendet werden.	3	Ja	Mittels Androidanbindung konnte dies Realisiert werden.
12	App-Info anzeigen	2	Ja	Informationen zum App in der „Home Activity“
13	Navigation mit Handy-Buttons (Back button, Touch etc.)	2	Ja	Mittels Androidanbindung konnten diese Events abgefangen und verarbeitet werden.
14	Nach dem Versenden einer Nachricht gelangt man wieder zur „Home Activity“	1	Ja	
15	App kann via Android-Store heruntergeladen werden	3	Ja	Im Play Store ersichtlich und gratis downloadbar!
16	Ausdruck der Nachricht (Print) auf Drucker	2	Zum Teil	Nachricht wird noch nicht an einen Drucker gesendet. Wird in neuem Release vorgesehen.
17	Kamera in MMS / Email aktivieren. Um direkt vom App ein Bild zu schießen und zu versenden	3	Nein	Nächster Release


## 9. Installationsanleitung Android

Die vollständige Installationsanleitung für das Android SDK in Eclipse ist auf folgender Seite zu finden: <http://developer.android.com/sdk/installing/installing-adt.html>

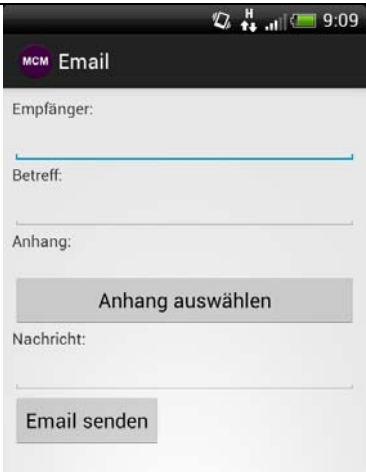
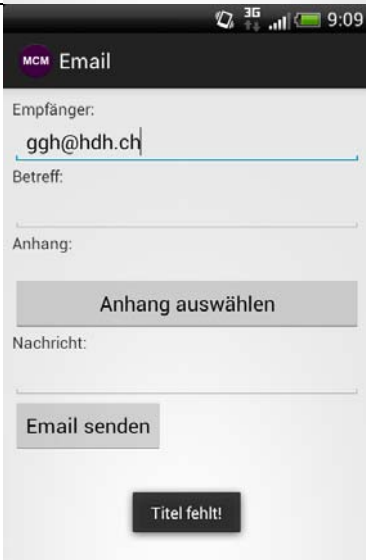
## 10. Benutzer Manual

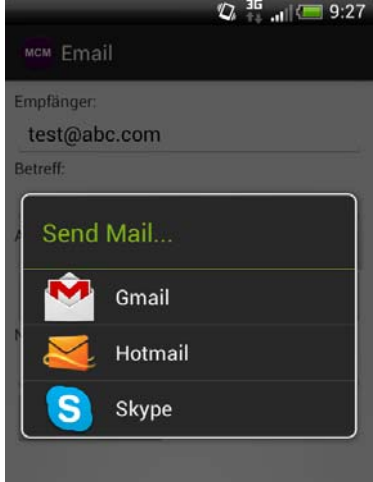
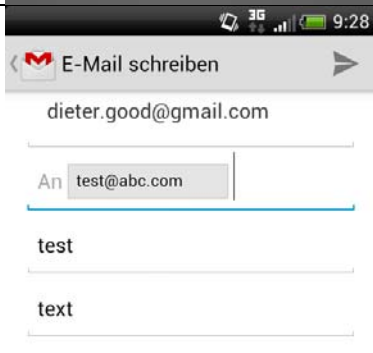

### 10.1 App herunterladen / Programmstart

#	Beschreibung	Grafik
1.	<p>Das App kann im Google Play Store heruntergeladen werden.</p> <p>Suchen nach: „MCM Multi Channel Mobile“</p>	 <p>10.01.2013 165 KB</p> <p><b>BESCHREIBUNG</b></p> <p>Dieses App ist im Rahmen einer Gruppensemesterarbeit im Herbstsemester 2012 anhand eines Lernauftrages in den</p>
2.	<p>Nach erfolgreicher Installation finden Sie das App in ihrem Apps-Menü.</p>	

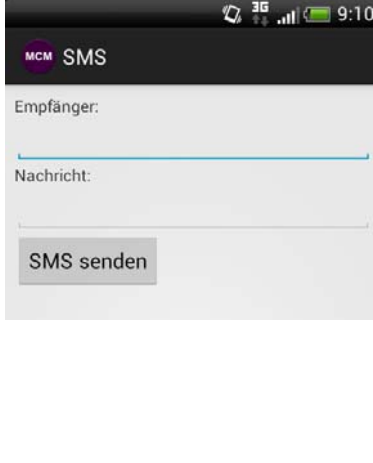
3.	<p>Bei Programmstart gelangen Sie automatisch zur Nachrichtenauswahl im MCM Home.</p> <p>In den nächsten Unterkapiteln werden die einzelnen Menüpunkte genauer beschrieben.</p>	
----	---	--

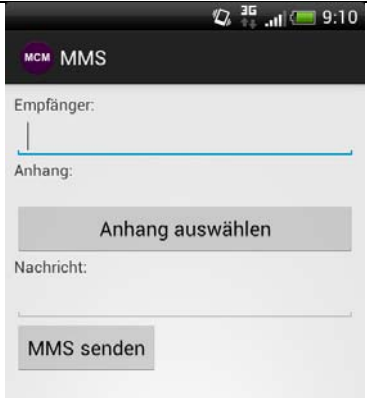

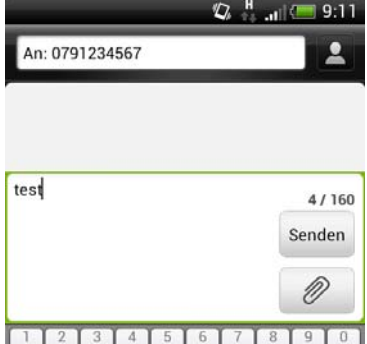

## 10.2 Email senden

4.	<p>Wenn Sie im MCM Home auf „Email senden“ klicken, gelangen Sie zur Eingabemaske.</p> <p>Geben Sie den Empfänger, Betreff und die Nachricht ein. Optional können Sie eine Datei als Anhang auswählen, die mitgesendet werden soll. Dies kann durch klicken auf „Anhang auswählen“ getan werden. Dabei öffnet sich der Android interne Filebrowser, mit welchem die Datei ausgewählt werden kann.</p>	
5.	<p>Bei Formatfehlern in der Email-Adresse (XX@yy.ch), oder bei fehlendem Betreff oder Nachrichtbody, erscheint beim Sendeversuch jeweils eine Fehlermeldung wie analog zur Abbildung.</p> <p>Erst wenn alle Pflichtfelder ausgefüllt sind, können Sie die Nachricht versenden.</p>	


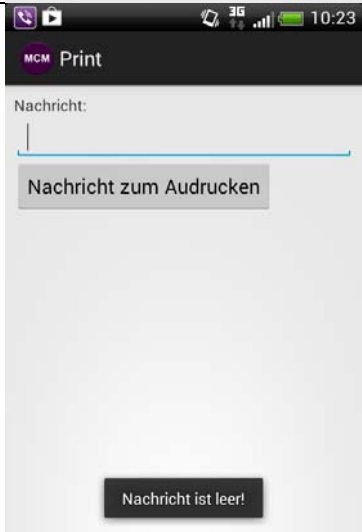
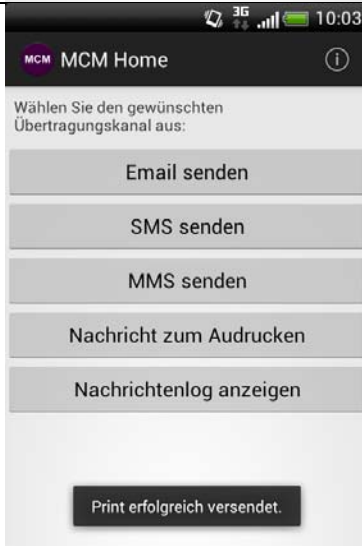
6.	<p>Beim betätigen des Buttons „Email senden“ wird automatisch der Android-Email-Client gestartet und die Nachricht zur Versendung vorbereitet.</p> <p>Hinweis: Falls mehrere Email-Clients installiert sind, wählen Sie den gewünschten aus.</p>	
7.	<p>Die zu versendete Nachricht wird nochmals im gewählten Email-Client dargestellt, und kann definitiv abgeschickt werden.</p>	
8.	<p>Nach dem Versenden der Nachricht gelangen Sie mit dem gerätspezifischen „Zurück-Button“ wieder auf die Startseite des MCM.</p>	

### 10.3 SMS senden / MMS senden

9.	<p>Wenn Sie im MCM Home auf „SMS senden“ klicken, gelangen Sie zur Eingabemaske.</p> <p>Geben Sie den Empfänger, Betreff und die Nachricht ein. Als Empfängerdaten können nur Ziffern (0-9) eingegeben werden.</p>	
----	--	--



	Wenn Sie ein MMS versenden, können Sie optional zusätzlich einen Anhang mitgeben.	
10.	<p>Fehlt beim Senden der Empfänger oder der Nachrichtenbody erscheint eine Fehlermeldung.</p> <p>Erst wenn alle Pflichtfelder ausgefüllt sind, können Sie die Nachricht versenden.</p>	
11.	<p>Beim betätigen des Buttons „SMS senden“ bzw. „MMS senden“ wird automatisch der Android-SMS-Client gestartet und die Nachricht zur Versendung vorbereitet.</p> <p>Klicken Sie erneut auf „Senden“, um die Nachricht endgültig zu versenden.</p>	
12.	Nach dem Versenden der Nachricht, gelangen Sie mit dem gerätspezifischen „Zurück-Button“ wieder auf die Startseite des MCM.	

## 10.4 Nachricht zum Ausdrucken

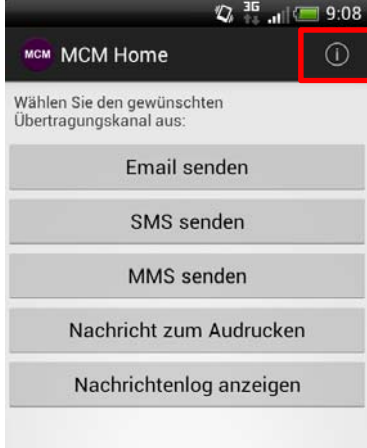
13.	Wenn Sie im MCM Home auf „Nachricht zum Ausdrucken“ klicken, gelangen Sie zur Eingabemaske.	 <p>The screenshot shows the 'MCM Print' screen. At the top, there's a status bar with '3G' and '9:10'. Below the title 'MCM Print', there's a label 'Nachricht:' followed by a text input field. At the bottom, there's a button labeled 'Nachricht zum Ausdrucken'.</p>
14.	<p>Fehlt beim Senden der Nachrichtenbody erscheint eine Fehlermeldung.</p> <p>Erst wenn alle Pflichtfelder ausgefüllt sind, können Sie die Nachricht auf den Printer senden.</p>	 <p>The screenshot shows the 'MCM Print' screen. At the top, there's a status bar with '3G' and '10:23'. Below the title 'MCM Print', there's a label 'Nachricht:' followed by a text input field. At the bottom, there's a button labeled 'Nachricht zum Ausdrucken'. A black error message box at the bottom center says 'Nachricht ist leer!'.</p>
15.	<p>Beim erfolgreichen versenden der Nachricht gelangen Sie automatisch wieder zur Startseite des MCM.</p> <p>Hinweis: Der Ausdruck auf ein Endgerät wurde noch nicht ganz ausprogrammiert, bzw. es fehlt der Drucker-Client.</p>	 <p>The screenshot shows the 'MCM Home' screen. At the top, there's a status bar with '3G' and '10:03'. Below the title 'MCM Home', there's a label 'Wählen Sie den gewünschten Übertragungskanal aus:'. Below this, there's a list of buttons: 'Email senden', 'SMS senden', 'MMS senden', 'Nachricht zum Ausdrucken', and 'Nachrichtenlog anzeigen'. At the bottom, there's a black confirmation message box that says 'Print erfolgreich versendet.'</p>



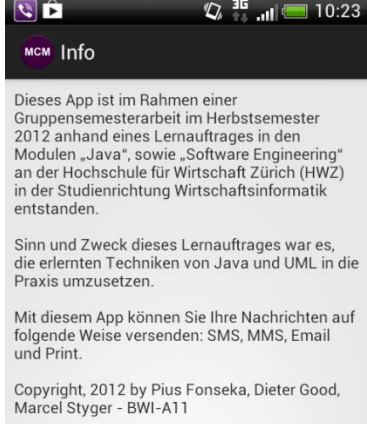
## 10.5 Nachrichtenlog anzeigen

16.	<p>Wenn Sie im MCM Home auf „Nachrichtenlog anzeigen“ klicken, gelangen Sie zur Übersicht.</p> <p>Im Log sind alle erfolgreich versendeten Nachrichten nach Datum sortiert.</p> <p>Der Aufbau ist wie folgt:</p> <p><i>Typ: Nachrichttyp (SMS / Email / MMS / Print)</i>  <i>Time: Absenderzeit und Datum</i>  <i>Nachricht: Nachrichtinhalt</i></p>	
17.	<p>Mit dem gerätspezifischen „Zurück-Button“ gelangen Sie wieder ins Hauptmenü des MCM.</p>	

## 10.6 Info anzeigen

18.	<p>Wenn Sie im MCM Home oben Rechts auf das Info-Symbol klicken, gelangen Sie zur Anzeige.</p>	
-----	--	--



19.	In der Info stehen alle Erläuterungen zum App.	
-----	--	--

## 11. Schlusswort

### 11.1 Fazit

Diese Projektarbeit hat uns aufgezeigt, wie aufwendig es sein kann, eine funktionsfähige Java-Applikation zu schreiben und vollumfänglich zu dokumentieren. Von der seriösen Erstellung des Projektplanes, bis hin zu den UML-Diagrammen und dem eigentlichen Programmieren, mussten wir uns immer wieder mit komplexen Themen auseinandersetzen. In erster Linie galt es, uns für eine Variante zu entscheiden, hinter der jedes Projektmitglied stehen konnte. Viele Ideen standen im Raum und davon musste ein Paket zusammengestellt werden, was auch wirklich realisierbar war. Man muss beachten, dass das Know-how in Software Engineering und Programmierung bei 2 von 3 Projektmitgliedern sehr begrenzt war.

Am schwersten fiel uns die Erstellung der geforderten UML-Diagramme, insbesondere die Aktivitäts- und Sequenzdiagramme. Doch nach gründlichen Recherchen und Gesprächen mit den Dozierenden konnten wir auch diese Hürde meistern.

Das Programmieren fiel uns am leichtesten, da wir innerhalb der Gruppe sehr gut miteinander harmonierten und uns gegenseitig unterstützten. Als Vorteil kam sicherlich dazu, dass mit Marcel Styger ein Android-Experte im Team war und den Neulingen stets zur Seite stand.

Die Anbindung unserer Lösung an das Android Betriebssystem stellte für uns einen ganz besonderen Reiz dar und gibt – vor allem den eher „Programmierneulingen“ – die Gewissheit, dass nichts unmöglich ist.

Mit Stolz schauen wir auf unsere vollbrachte Arbeit, in der wir viel über die Materie der Programmierung und Modellierung lernen konnten. Alle gestellten Anforderungen wurden umgesetzt und der OO-Ansatz mit der zusätzlichen Erweiterungsmöglichkeit des Apps wurde ebenso miteinbezogen.

## 11.2 Lernbericht je Teilnehmer

### 11.2.1 Pius Fonseca

Der spannendste Teil war für mich das Erstellen des Use Cases. Diese Arbeit bot mir am einfachsten Zugang in das Verständnis des Programmierens (Abfolgen-, Verknüpfungen und Abhängigkeiten der Use Cases zueinander wie auch die klare Abgrenzung der Aufgaben der unterschiedlichen Use Cases). Als Programmierneuling habe ich erkannt, dass ein kompletter Projektablauf hin bis zur Realisierung des Programms und der Fertigstellung der Dokumentation sehr viel „Handarbeit“ und aktives, kreatives Denken erfordert. Obwohl mit der Projektarbeit sehr viel Theorie verbunden ist, verlangt die Erstellung viel praktisches, angewandtes Denken. Letzteres ist oftmals die grösste Herausforderung wie mir oft schien.

Durch das Know-how und die Fähigkeiten der anderen Gruppenmitglieder habe ich besonders dadurch profitiert, dass ich nicht alle Bausteine von Grund auf selbst erarbeiten musste. Dennoch erscheint mir das effektive Programmieren und die Erstellung der UML-Diagramme immer noch recht abstrakt.

Meine wichtigste Erkenntnis während des Projekts war, zu erfahren, wie wichtig es ist Kommunikationsvereinbarungen im Team einzuhalten. Bei Abweichungen oder neuen Ideen musste jeweils die Gruppe konsultiert werden. Die Art der Zusammenarbeit in der Gruppe erwies sich als sehr effizient in Hinblick auf unser Ziel. Insbesondere das ausgeprägte Know-how des „Gruppenleiters“ machte es möglich, nicht auf Abwege zu geraten und immer mehr oder weniger direkt dem Ziel entgegen zu arbeiten.

### 11.2.2 Dieter Good

Obwohl ich bereits in der Grundbildung als Informatiker ein wenig Programmieren erlernt hatte, war dieser Auftrag etwas ganz neues für mich. Ich hätte nie gedacht, dass ich in nächster Zeit einmal ein Android-App entwickeln werde, da für mich die Softwareentwicklung zu einem der Themen gehört, um die ich lieber einen Bogen mache. Daher war der Respekt vor diesen Modulen und dem eigentlichen Lernauftrag auch dementsprechend gross. Die Anfangsängste legten sich jedoch nach den ersten paar Java-Lektionen und das Interesse am Entwickeln übernahm die Überhand.

Die Zusammenarbeit innerhalb des Projekts mit meinen zwei Studienkollegen war äusserst angenehm und wir konnten unsere gesetzten Ziele schnell erreichen. Dank Marcel Styger hatten wir einen absoluten Android-Profi im Team, was uns das ganze natürlich massiv vereinfachte.

Nach dem sammeln und Ausdiskutieren der Ideen, erstellten wir das Klassendiagramm und wagten uns an den Code. Nachdem M. Styger das Java-Grundgerüst anfertigte, konnten P. Fonseca und ich auch unseren Teil am Code beitragen, und programmierten einige Methoden und Klassen aus. Das Zusammenführen der einzelnen Codes und die Konvertierung in Android wurden dann wieder von M. Styger übernommen. Im Bereich der UML hatten wir am Anfang gewisse Probleme, unsere Arbeit in die entsprechenden Modelle zu übertragen, da wir meist zu weit gedacht hatten. Doch dank gezielten Inputs der Dozenten kamen wir mit guter Teamarbeit anschliessend auf das gewünschte Resultat. Ich war zudem für die Dokumentation und dem Einhalten der gesetzten Anforderungen zuständig.

Ich bin sehr stolz auf unser vollbrachtes Werk und habe mich auch mit den Teamkameraden bestens verstanden.

Abschliessend kann ich sagen, dass ich zwar auch heute noch lieber anderen den Vorrang lasse, wenn es darum geht etwas zu programmieren, aber dank dieses Lernauftrags die Distanz zu diesem Thema für mich erheblich kleiner geworden ist.

### 11.2.3 Marcel Styger

Als Software – Entwickler ist man sich gewohnt, fertige Anforderungen zu erhalten und oftmals gleich „los programmieren“ zu können – nicht jedoch in dieser Arbeit. Eine App von Grund auf neu zu erstellen verlangt konzeptionelle Vorarbeiten. So mussten wir uns zuerst überlegen, was die App überhaupt kann (Use Cases). Erst durch das schematische Skizzieren des Klassendiagramms wurde uns nach und nach bewusst, was getan werden kann und wo Optimierungen gemacht und Synergien in den Klassen genutzt werden können.

Vor allem herausfordernd war für mich die reibungslose Integration aller Code-Fragmente, welche wir nach und nach programmierten und über Git-Hub geteilt haben. Weiter war es für mich eine Herausforderung, meinen Teamkameraden die Android - Entwicklung erklären zu können. Es ist natürlich nicht trivial, als Java - Einsteiger gleich eine vollumfängliche App zu kreieren.

Als neuer Lerninhalt verstand ich das richtige Erstellen eines Klassendiagramms. Selbstverständlich habe ich in meinem beruflichen Weg als Softwareentwickler dutzende solcher Diagramme erstellt. Diese waren jedoch von eher primitivem Charakter, wenn ich sie so mit unserem Diagramm vergleiche, vor allem was die Präzision der Deklarationen und der Gleichen angeht. Auch die Sequenzdiagramme sind im Android - Kontext nicht ganz einfach zu erstellen. So gibt es zahlreiche Aufrufe, welche die Objekte während ihres Lebenszyklusses machen. Wir haben uns darauf geeinigt, nur die für unsere Umgebung und die unser Klassenmodell tangierenden Objekte bzw. Methodenaufrufe abzubilden.

Auch war es für mich sehr spannend, mit zwei Kameraden unserer Klasse zusammenzuarbeiten. Eine Gruppenarbeit lehrt einen auch viel über das Bilden einer Gruppe sowie die Zusammenarbeit zwischen verschiedenen Menschen mit unterschiedlichen Stärken.

Für mich ist die Softwareentwicklung für mobile Endgeräte sehr Zukunftsträchtig. Es freut mich, die Möglichkeit bekommen zu haben ein Stück weit Forschungen betreiben zu können in diesem Bereich, im Rahmen dieses Projektes.

## 12. Abbildungsverzeichnis

Abbildung 1: Projektplan.....	5
Abbildung 2: UC Diagramm .....	7
Abbildung 2: Sequenzdiagramm Nachricht senden .....	9
Abbildung 3: Aktivitätsdiagramm Nachricht senden.....	10
Abbildung 4: Sequenzdiagramm Log anzeigen .....	12
Abbildung 5: Sequenzdiagramm Info anzeigen .....	13
Abbildung 6: Klassendiagramm.....	14
Abbildung 7: Android Activity Life Cycle, Quelle: <a href="http://developer.android.com">http://developer.android.com</a> .....	16