Projektkatalog

für das Softwareprojekt

GPS Anwendung für mobile Endgeräte

Ein Projekt von

Philipp Lehmann (Matrikelnr.: 17002)

Inhaltsverzeichnis

1 Project Brief

1.1 Hintergrund

Im Rahmen der Projektarbeit (21902) soll eine Applikation für mobile Endgeräte erstellt werden, welche das Senden von GPS-Koordinaten ermöglicht. Diese Anwendung soll auf Blackberry-Geräten mit integriertem GPS-Empfänger zum Einsatz kommen. Der Initiator dieses Projektes ist die Firma

Redcoal Pty. Ltd

45 Lime Street, King Street Wharf

Sydney 2000 Australia

Das Projekt soll in diesem Unternehmen umgesetzt und eingeführt werden. Die Entwicklung findet durch Philipp Lehmann statt. Design und Konzeption des Projektes werden in Absprache mit dem Unternehmen gemeinsam gelöst.

1.2 Projektdefinition

1.2.1 Projektziel

Ziel des Projektes ist es, eine Applikation für mobile Endgeräte zu erstellen, welche in Intervallen Positionskoordinaten sendet. Weiterhin soll eine Option in der mobilen Anwendung bereitgestellt werden, um einen Arbeitsbeginn und ein Arbeitsende zu senden. Als Programmiersprache soll für dieses Projekt J2ME zum Einsatz kommen. Ferner soll die Applikation für mobile Endgeräte für den Gerätetyp Blackberry geschrieben werden. Falls es die Projektzeit ermöglicht, soll diese Software auch für Nokia-Geräte bereitgestellt werden.

Redcoal Pty. Ltd ortet derzeit mobile Endgeräte ausschließlich über den Weg der Triangulation. Da dieses Verfahren eine Einschränkung gegenüber dem gewähltem Carrier und der Genauigkeit der Ortung bedeutet, soll das Projekt dem Unternehmen eine weitere wichtige Möglichkeit bieten, mobile Endgeräte zu orten und auf seiner existierenden Location Based Service Anwendung „Simpoint“ bereitzustellen. Das Ziel ist es, neue Kunden zu werben und bestehenden Kunden eine weitere, exaktere Option zu bieten, um ihre Mitarbeiter zu lokalisieren.

1.2.2 Projektumfang

Das Projekt ist im Wesentlichen in zwei Bereiche zu unterteilen. Die Applikation, welche auf dem mobilen Endgerät läuft, zählt zu dem ersten Bereich. In dem zweiten Bereich wird die Anwendung eingeordnet, welche die Koordinaten erhält, ausliest und in die Datenbank speichert. Die Übertragung erfolgt hierbei für den Blackberry-Gerätetyp via E-Mail. Die Übertragung über E-Mail wurde gewählt, da Blackberry diesen Service kostenfrei zur Verfügung stellt und eine unbegrenzte Anzahl an E-Mails gesendet werden können.

Die Applikation auf dem mobilen Endgerät soll mit J2ME erstellt werden. Da auf diesem Fachgebiet bisher kaum Kenntnisse vorhanden sind, muss hierfür eine gewisse Einarbeitungszeit einkalkuliert werden. Die Anwendung auf dem Server, welche für das Lauschen, Auslesen und Speichern verantwortlich ist, wird mit der von Microsoft angebotenen Programmiersprache C# umgesetzt. Auf diesem Gebiet bestehen durch meine Praktikumstätigkeit ausreichend Kenntnisse, so dass dieser Teilbereich zügig umgesetzt werden kann. Das gesamte Projekt, einschließlich Test und der anschließenden Projektdokumentation, soll innerhalb von 480 Stunden realisiert werden.

1.2.3 Projektansatz

Wie bereits im Abschnitt 1.2.2 erwähnt, besteht das Projekt im Wesentlichen aus zwei Teilbereichen. Der erste Teilbereich umfasst die Entwicklung der J2ME-Anwendung auf dem mobilen Endgerät. Der zweite Teilbereich beinhaltet die Erstellung einer Software zum Lauschen, Auslesen und Speichern der Koordinaten in eine Datenbank. Zu Beginn soll der erste Teilbereich umgesetzt werden. Dieser beinhaltet das Empfangen und das anschließende Senden der Koordinaten. Das Empfangen und Senden soll in einem von dem Kunden bestimmbaren Intervall erfolgen. Hierfür soll eine Programmoberfläche entwickelt werden, die dem Kunden die Optionen bietet, die Ortung zu beenden und zu starten. Für das Versenden des Arbeitsbeginns und des Arbeitsendes sollen zwei Button „ClockIN“ und „ClockOut“ erstellt werden. Weiterhin soll es dem Kunden möglich sein, eine Zeit und eine Auftragsnummer an die Nachricht anzufügen. Bei der Erstellung des Klassen-Designs soll darauf geachtet werden, die Anwendung möglichst geräteneutral zu halten, um möglichst viel Programmcode für andere Gerätetypen verwenden zu können. Alle Aufgabenteile werden von dem Projektantragsteller allein umgesetzt.

Wichtige Aspekte für die Projektansatzentscheidung:

Beeinflussende Faktoren	**Erläuterung**
Langfristige Strategie	Entwicklung einer Software zur Ortung von mobilen Endgeräten mit GPS-Funktionalität.
Infrastruktur	Ein PC, drei Blackberry-Geräte (ein 8800 mit GPS-Funktionalität, 8707g und 7130g)
Zur Verfügung stehende Mittel	Kommerzielles Projekt. Etwaige Geräte (Nokia Mobile-Phone) und Gebühren können erworben werden.
Einschränkung der Zeit	Die praktische Umsetzung des Projektes soll in dem Unternehmen erfolgen. Hierfür ist eine Zeitspanne von 10 Wochen vorgesehen. Die Dokumentation wird in Deutschland fertiggestellt. Hierfür wird ein Zeitraum von zwei Wochen einkalkuliert.
Risikoerwägung	Das Projekt kann nicht innerhalb der zehn Wochen umgesetzt werden. Zielsetzung kann durch technische Schwierigkeiten und falsche Planung nicht erreicht werden Schwierigkeiten bei der Einarbeitung in J2ME und der zu verwendenden Blackberry-Bibliotheken Anwendung benötigt zu viele Ressourcen (Batterieverbrauch)
Anforderung an die Ausbildung	Erweiterung der Java-Kenntnisse. Umgang mit J2ME. Erweiterung von Kenntnissen im Bereich MS C# und MS SQL.
Sicherheitsanforderung	Der Benutzer soll erstmalig über seine E-Mail-Adresse identifiziert werden. Weitere Identifikation erfolgt über die IMEI-Nummer des Gerätes. E-Mail-Abfrage soll über SSL erfolgen.
Anforderungen an Qualität	Die erstellte Applikation soll auf jedem Blackberry-Device mit integriertem GPS-Empfänger lauffähig sein. Das Programm soll einfach, verständlich und benutzerfreundlich sein.
Eingesetzte Software	Frei erhältliche JDE für Blackberry-Devices von Research In Motion, sowie Blackberry-Simulator 8800. Microsoft Visual Studio 2005 und Microsoft SQL Server 2000.
Erwartete Dauer	Dauer der Umsetzung: 3 Monate
Projektübergaben	25.06.2008 (Presentation Day) bzw. 26.06.2008 (Media Night).

1.2.4 Projektmanagement Team

Das Projektmanagement Team besteht aus folgender Person:

Rolle	Name	Telefon	E-Mail
Projekt-Manager	Philipp Lehmann	0179/546 564 1	mail@philipplehmann.de

1.2.5 Voraussetzungen

Um die Applikation auf einem Blackberry-Phone benutzen zu können, benötigen wir eine Lizenz von Blackberry. Diese muss bei Blackberry direkt erworben werden.

1.3 Business Case

Das Projekt soll nach Fertigstellung an Kunden bereitgestellt werden. Da eine enge Partnerschaft mit dem Mobilfunkanbieter Optus besteht, sollen ebenfalls Optus-Kunden dieses Produkt erhalten. Über die entsprechenden Einnahmen stehen mir keine Informationen zur Verfügung.

Eigener Nutzen

- Erwerb neuer Kenntnisse
- Erfahrung mit der Erstellung und Organisation eines Projektes
- Benotung und Leistungsergebnis
- Ein vorzeigbares Projekt, welches präsentiert werden kann

1.4 Qualitätserwartung

Folgende Qualitätsanforderungen sind zu erfüllen:

- Hohe Stabilität: Langzeittest ohne Programmabstürze
- Usability: Das Programm soll schnell, verständlich und einfach zu handhaben sein
- Die Applikation soll möglichst auf allen Geräten des jeweiligen Handytyps mit integriertem GPS-Empfänger lauffähig sein.
- Sicherheit: Da Blackberry bereits ein sicheres Konzept zum Versenden der E-Mail anbietet, soll die Kommunikation zwischen Mailprovider und Pop3MailClient über SSL laufen.

1.5 Abnahmekriterien

Folgende Abnahmekriterien sind für die Abnahme zu erfüllen:

- Vorhandensein aller Funktionen
- Software muss auf mobilen Endgeräten lauffähig sein
- Die Benutzerfreundlichkeit muss eingehalten werden
- Koordinaten sollen exakt und in bestimmbarem Intervall gesendet werden
- E-Mail-Listener muss Koordinaten korrekt auslesen und in Datenbanken speichern
- E-Mail-Listener muss dauerhaft und sicher auf dem Server laufen.

1.6 Risiken

Folgende Risiken wurden für das Projekt identifiziert:

- Das Projekt kann nicht in den zeitlichen Vorgaben fertiggestellt werden
- Vorgaben können aufgrund von fehlenden Kenntnissen nicht in der gewünschten Art und Weise implementiert werden
- Die Übertragungstechnik ist für diese Applikation ungeeignet
- Schwierigkeiten bei der Einarbeitung der Software (Research In Motion). Dadurch könnten ebenfalls Verzögerungen auftreten.
- Das Blackberry-Phone ist für das dauerhafte und häufige Senden von GPS-Koordinaten ungeeignet (schlechter GPS-Empfänger)

2 Projektplan

Das gesamte Projekt soll in zwei Bereiche gegliedert werden.

- Erstellen der Applikation auf dem mobilen Endgerät
- Erstellen des E-Mail-Clients zum Auslesen und Speichern der Koordinaten

2.1 Erstellen der Applikation auf dem mobilen Endgerät

In diesem Teilbereich des Projektes soll die Applikation für das mobile Endgerät erläutert werden. Dieser Teilbereich gliedert sich in drei Unterbereiche:

- Erstellen eines Dienstes zum Empfangen von GPS-Koordinaten
- Erstellen eines E-Mail-Versenders
- Entwicklung einer API

2.1.1 Erstellen eines Dienstes zum empfangen von GPS-Koordinaten

Dieser Teilbereich der Kategorie „Erstellen der Applikation auf dem mobilen Endgerät“ beinhaltet das Erstellen eines Dienstes, welcher in Intervallen auf GPS-Koordinaten lauscht, sie empfängt und zur Versendung in einem String speichert. Hierbei soll das Intervall von dem Benutzer selbst bestimmt werden können. Weitere Optionen sollen in diesem Teil das Stoppen und das Starten des Dienstes (des Lauschens) sein. Dem Benutzer soll zusätzlich die Signalempfangsstärke angezeigt werden. Werden keine Koordinaten durch schlechte äußere Bedingungen empfangen, soll ein Signal-Status auf den Wert „Bad“ gesetzt werden. Bei Erhalt von GPS-Koordinaten soll dieser auf den Signal-Status „Good“ wechseln.

2.1.2 Erstellen eines E-Mail-Versenders

Der E-Mail-Versender soll die vom GPS-Listener bereitgestellten Informationen an eine gewünschte E-Mail-Adresse versenden. Weiterhin sollen die E-Mail-Nachrichten nach dem Versand aus der Outbox des Mobile-Phone gelöscht werden. Zu versendende Attribute sind hier

- Longitude

- Latitude
- Altitude
- Speed
- Time
- Standard-E-Mail-Adresse des Benutzers
- IMEI-Nummer des Gerätes

2.1.3 Entwicklung einer API

Die zu erstellende Benutzeroberfläche soll benutzerfreundlich und einfach zu handhaben sein. Das Hauptmenü der API soll sich in folgende Bereiche untergliedern:

- Menüpunkt Simpoint:
- Menüpunkt Clock IN/OUT
- Menüpunkt About

2.1.3.1 Menüpunkt Simpoint:

In diesem Menü soll dem Benutzer die Möglichkeit geboten werden, das Empfangen und Senden von Koordinaten zu stoppen und zu starten.

2.1.3.2 Menüpunkt Clock IN/OUT:

In diesem Menü können der Arbeitsbeginn und das Arbeitsende gesendet werden. Weiterhin können eine Zeit und eine Job-ID angegeben werden, die in der E-Mail versandt werden sollen.

2.1.3.3 Menüpunkt About:

In dem Menü „About“ erfährt der Benutzer Informationen über das Unternehmen. Inhalt sollen hier die Kontaktinformationen des Unternehmens sein.

2.2 Erstellen des E-Mail-Client

Um die von der Applikation gesendeten Informationen auszuwerten und weiter verarbeiten zu können, benötigen wir einen E-Mail-Client, welcher die E-Mail-Nachrichten von einem E-Mail-Provider über Pop3 abholt und sie danach auswertet, um sie im nächsten Schritt in einer Datenbank zu speichern. Der E-Mail-Client soll in einem bestimmbaren Intervall E-Mail-Nachrichten von dem E-Mail-Provider abholen.

Die Verbindung erfolgt hierbei über die sichere Übertragung Secure Sockets Layer (SSL). Die Aufgabenstellung soll in drei Bereiche unterteilt werden.

- Erstellen des E-Mail-Client
- Speichern der empfangenen Information in die Datenbank
- Stored Procedure in der Datenbank

2.2.1 E-Mail-Client

Der E-Mail-Client hat die Aufgabe, sich bei dem E-Mail-Provider anzumelden, die eingegangenen E-Mails herunterzuladen und sich danach wieder auszuloggen. Die Abfrage nach E-Mail-Nachrichten soll in einem Intervall von ein bis zwei Minuten erfolgen. Sobald E-Mail-Nachrichten empfangen wurden, sollen diese einzeln an eine Klasse StoreCoordinates weitergegeben werden, welche die Koordinaten in die Datenbank speichert. Die Klasse StoreCoordinates soll im folgenden Punkt 2.2.2 näher erläutert werden.

2.2.2 Speichern der empfangenen Informationen

Die Klasse StoreCoordinates hat die Aufgabe, die Informationen zu separieren und auszuwerten. Danach soll geprüft werden, ob die gesendeten Informationen zu einem bestehenden und aktiven Kunden, sowie zu einem von ihm angemeldeten Mobile-Phone zuzuordnen sind. Als aktiv gilt ein Kunde, wenn er zum Beispiel die monatlichen Gebühren bezahlt hat. Nur wenn die eingegangenen Informationen einem Gerät zuzuordnen sind, werden sie, wie im folgenden Schritt betrachtet, in der Datenbank gespeichert.

2.2.3 Stored Procedure in der Datenbank

Um die Datenbanktransaktion effizient zu gestalten, sollen für dieses Projekt Stored Procedure zum Einsatz kommen. Diese sollen die Koordinaten in der Datenbank speichern. Die Wahl für Stored Procedure erfolgte, um mehrere Abfragen zu verbinden und den Daten-Traffic durch eine Minimierung der zu übertragenden Daten, zu verringern.

3 Zeitplanung und Milestones

Das Projekt soll innerhalb von zehn Wochen fertiggestellt werden. Damit dieser Zeitplan eingehalten werden kann, müssen Milestones gesetzt werden, um das Projekt zeitlich zu überwachen und zu koordinieren.

Milestone	**Zeitplanung**	**Erläuterung**
Installieren der Blackberry-Anwendung und des Blackberry-Simulator	Ca.: 3 Tage	Herunterladen der Software und des geeigneten Simulators. Einarbeitung in das Programm.
Erstellen des GPS-Listener.	Ca.: 2 Wochen	Erstellen des Dienstes, welcher die GPS-Koordinaten empfangen und herauslesen soll
Erstellen des E-Mail-Versenders	Ca.: 1 Woche	Erstellen der Klasse, welche für das Senden der E-Mails zuständig ist. Die Klasse soll außerdem das Löschen der gesendeten E-Mails übernehmen
Erstellen der Benutzeroberfläche	Ca.: 2 Wochen	Die Benutzeroberfläche soll dem Anwender die im Punkt 2.1.3 beschriebenen Funktionen zur Verfügung stellen.
Erstellen des E-Mail-Clients	Ca.: 2 Wochen	Dieser soll eine Pop3-Verbindung öffnen und die E-Mail-Nachrichten vom E-Mail-Provider herunterladen
Speichern der Koordinaten mithilfe von Stored Procedure in die Datenbank	Ca.: 2 Wochen	Dieser Teil der Entwicklung soll die Informationen auswerten und sie in die Datenbank speichern.
Testphase	Ca.: 3 Tage	Test der erstellten Anwendungen
SUMME	**Ca: 10 Wochen**	

4 Statusreport

4.1 Zeitplanung und Milestones

Die in dem Projekt zu erfüllenden Milestones werden im Folgenden noch einmal aufgezeigt und beschrieben. Sie sollen als Grundlage für den Statusreport dienen.

Nr.	Milestone	Zeitplanung	Erläuterung
1	Installieren der Blackberry-Anwendung und des Blackberry-Simulator	Ca.: 3 Tage	Herunterladen der Software und des geeigneten Simulators. Einarbeitung in das Programm.
2	Erstellen des GPS-Listener.	Ca.: 2 Wochen	Erstellen des Dienstes, welcher die GPS-Koordinaten empfangen und herauslesen soll
3	Erstellen des E-Mail-Versenders	Ca.: 1 Woche	Erstellen der Klasse, welche für das Senden der E-Mails zuständig ist. Die Klasse soll außerdem das Löschen der gesendeten E-Mails übernehmen
4	Erstellen der Benutzeroberfläche	Ca.: 2 Woche	Die Benutzeroberfläche soll dem Anwender die im Punkt 2.1.3 beschriebenen Funktionen zur Verfügung stellen.
5	Erstellen des E-Mail-Clients	Ca.: 2 Wochen	Dieser soll eine Pop3-Verbindung öffnen und die E-Mail-Nachrichten vom E-Mail-Provider herunterladen
6	Speichern der Koordinaten mithilfe von Stored Procedure in die Datenbank	Ca.: 2 Wochen	Dieser Teil der Entwicklung, soll die Informationen auswerten und sie in die Datenbank speichern.
7	Testphase	Ca.: 3 Tage	Test der erstellten Anwendungen
	SUMME	**Ca: 10 Wochen**	

4.2 Statusbericht

Für den Projektbetreuer wird regelmäßig ein Statusbericht zusammengestellt, welcher den aktuellen Fortschritt des Projekts dokumentiert. Dies soll jeweils im Abstand von zwei Wochen erfolgen. Sollten besondere Situationen es erforderlich machen, können diese Abstände entsprechend verkürzt werden. Der Statusbericht bezieht sich auf die im Anforderungskatalog festgelegten und beschriebenen Arbeitsschritte. Er soll die einzelnen Tätigkeiten der Projektarbeit aufzeigen sowie die Dauer der Projektphasen festhalten. Dadurch können eventuelle Engpässe in der Projektplanung früher erkannt und Gegenmaßnahmen getroffen werden.

4.2.1 Projektbeschreibung

Im Rahmen der Projektarbeit (21902) soll eine Applikation für mobile Endgeräte erstellt werden, welches das Senden von GPS-Koordinaten ermöglicht. Diese Anwendung soll auf Blackberry-Geräten mit integriertem GPS-Empfänger zum Einsatz kommen. Der Initiator dieses Projektes ist die Firma

Redcoal Pty. Ltd

45 Lime Street, King Street Wharf

Sydney 2000 Australia

Das Projekt soll in diesem Unternehmen umgesetzt und eingeführt werden. Die Entwicklung findet durch Philipp Lehmann statt. Design und Konzeption des Projektes werden mit Absprache des Unternehmens gemeinsam gelöst.

4.2.2 Milestone eins

4.2.2.1 Beschreibung

Milestone	Installation der einzusetzenden Software
Zweck	Erstellen und Test der Anwendung auf dem Entwicklungsrechner
Abhängigkeiten	Keine
Erforderliche Qualitätsprüfung	Lauffähigkeit der Software. Richtige Wahl des Simulators

4.2.2.2 Tätigkeit

Um den festgelegten Projektplan einhalten zu können, wurde bereits eine Woche vor Semesterbeginn, Informationen über die einzusetzende Software recherchiert. Die

Informationen wurden direkt bei dem Hersteller gesammelt. Die Software Blackberry JDE 4.3.0, welche zur Umsetzung des Projektes erforderlich ist, wurde auf der Website

http://na.blackberry.com/eng/developers/downloads/jde.jsp

heruntergeladen. Um die entwickelte Software auf dem PC testen zu können, kommt der Blackberry Device Simulator 4.2.1.90 8800-O2 zum Einsatz. Er wurde gewählt, da dem Projektentwickler dieses mobile Endgerät zur Projektumsetzung zur Verfügung gestellt wird. Als nächster Schritt wurde versucht, ein Projekt zu erstellen und den einzusetzenden Simulator zu starten.

4.2.2.3 Aufgetretene Probleme und Lösungen

4.2.2.3.1 Erste Problemstellung

Problem :

Das erste Problem betraf das Erstellen eines Projektes. Durch das erstmalige Benutzen dieser Entwicklungsumgebung hatte ich anfänglich Schwierigkeiten, ein Projekt anzulegen.

Lösungsansatz:

Blackberry stellt auf seiner Internetseite Dokumentationen zu all seinen Entwicklungsumgebungen bereit. Hierbei half mir die Dokumentation BlackBerry_Application_Developer_Guide_Volume_1, die genau aufzeigt, wie ein Projekt anzulegen ist und welche Einstellungen beachtet werden müssen.

Lösung:

Um ein Projekt anzulegen, muss zuerst ein Workspace.jdw angelegt werden. Nach dem Anlegen eines Workspace, kann man in diesem ein Projekt erzeugen. Zu einem Projekt können beliebig viele Ressource-Dateien hinzugefügt werden.

4.2.2.3.2 Zweite Problemstellung

Problem:

Die zweite Problemstellung des Milestones eins betraf das Aktivieren des Projektes und der anschließende Einsatz des Simulators.

Lösungsansatz:

Wie in der ersten Problemstellung half mir hier die von Blackberry bereitgestellte Dokumentation weiter.

Lösung:

Ein Projekt muss vor dem Kompilieren und anschließendem Verwenden in einem Simulator aktiviert werden. Um anschließend den nachträglich installierten Simulator 4.2.1.90 einsetzen zu können, muss dieser zuvor in den Projektreferenzen festgelegt werden.

4.2.2.3.3 Dritte Problemstellung

Problem:

Versenden von E-Mail-Nachrichten auf dem einzusetzenden Simulator.

Lösungsansatz:

Um E-Mail-Nachrichten von einem Simulator versenden und empfangen zu können, stellt Blackberry einen E-Mail-Server Simulator (ESS) bereit. Hierbei stehen dem Entwickler zwei Möglichkeiten zur Verfügung:

- Standalone Mode
- Connected Mode

Lösung:

Erst nach längerem Test und Installation der Software Outlook Express ist es mir gelungen, E-Mail-Nachrichten zu versenden.

4.2.2.4 Qualitätskontrolle

Die zu verwendende Software ist lauffähig installiert worden. Alle Programmteile können ordnungsgemäß eingesetzt werden.

4.2.2.5 Risiken

Durch die aufgetretenen Problemstellungen hat sich die Umsetzung des ersten Teilbereiches um 2 Tage verlängert. Es wird versucht, in dem nächsten Teilbereich Zeit aufzuholen

4.2.2.6 Nächste Aufgabenstellung

Als nächste Aufgabenstellung gilt die Umsetzung des zweiten Milestone. Dieser beinhaltet die Entwicklung des GPS-Listener auf dem mobilen Endgerät.

4.2.3 Milestone zwei

4.2.3.1 Beschreibung

Milestone	Erstellen des GPS-Listener auf dem mobilen Endgerät
Zweck	Empfang der Koordinaten die zur Ortung des mobilen Endgerätes notwendig sind
Abhängigkeiten	Installation der Software auf dem Entwicklungsrechner
Erforderliche Qualitätsprüfung	Korrektheit der empfangenen Koordinaten. Einstellung eines Intervalls. Lauffähigkeit auf dem mobilen Endgerät

4.2.3.2 Tätigkeit

Zur Umsetzung dieses Milestone wird ein Projekt RedcoalAPI angelegt. Desweiteren wird eine Klasse GPSListener erstellt. Diese soll den Empfang der GPS-Daten gewährleisten. Hierfür wird eine innere Klasse LocationListenerImpl angelegt, welche das Interface LocationListener implementiert. Das Interface LocationListener fordert die Methode locationUpdated(LocationProvider, Location). Die Methode locationUpdated liest die Koordinaten aus und stellt sie bereit. Die Koordinaten werden mit location.getQualifiedCoordinates().getLongitude(); und getLatitude ausgelesen. Des weiteren können Altitude und Speed ermittelt werden. Die gesamten Daten speichere ich derzeit in einem String-Buffer um sie später an die Klasse Send-E-Mail übergeben zu können.

Das Interessante ist, das beim Anlegen des Location Provider. SetLocationListener ein Intervall angegeben werden kann. Dies ermöglicht das Empfangen der Koordinaten in einem bestimmbaren Zeitraum.

Weiterhin soll in diesem Milestone eine Testanwendung geschrieben werden, die den Erhalt der Koordinaten auf dem Blackberry-Device testet.

4.2.3.3 Aufgetretene Probleme und Lösungen

4.2.3.3.1 Erste Problemstellung

Problem:

Kein Empfang der Koordinaten. Erhalt der Exception LocationException“ und “Illegal-ArgumentException”.

Lösungsansatz:

Die LocationException wird ausgelöst, sobald die Methode LocationProvider.getInstance aufgerufen wird. In der RIM Device Java Libery erhielt ich Informationen über diese Exception. Sie wird aufgerufen, wenn kein LocationProvider zur Verfügung steht.

Lösung:

In dem Blackberry-Entwickler-Forum erfuhr ich, dass die Exception Location-Provider aufgerufen wird, wenn keine Koordinaten zur Verfügung stehen. Diese Koordinaten müssen vor Programmstart gesetzt werden. In dem eingesetzten Simulator findet man einen Menüpunkt GPS Location. Dort müssen GPS-Koordinaten eingetragen werden, die später von dem GPS LocationProvider empfangen werden.

4.2.3.4 Qualitätskontrolle

Die Koordinaten werden in Intervallen auf dem Simulator und dem Blackberry Device empfangen und in einer kleinen Test-API angezeigt. Die Anwendung prüft, ob die empfangenen Koordinaten valid oder not valid sind. Falls die Daten not valid sind, wird eine Fehlermeldung am Bildschirm ausgegeben. Die empfangenen Koordinaten werden in einem StringBuffer gespeichert. In diesem Milestone traten keine größeren Verzögerungen auf, so dass das Zeitlimit eingehalten werden konnte.

4.2.3.5 Nächste Aufgabenstellung

Als nächstes soll eine Testapplikation geschrieben werden, um den GPS-Empfänger auf dem Blackberry Device zu testen. Desweiteren soll der Milestone drei umgesetzt werden. Er beinhaltet die Umsetzung des E-Mail Versenders. Dieser soll die in diesem Milestone erhaltenen Koordinaten an eine beliebige E-Mail-Adresse versenden.

4.2.4 Milestone drei

4.2.4.1 Beschreibung

Milestone	Erstellen des E-Mail-Versenders
Zweck	Versenden der empfangenen Positionsdaten
Abhängigkeiten	Der im Milestone zwei entwickelte GPS-Listener gilt als Grundlage für diesen Milestone
Erforderliche Qualitätsprüfung	Positionsdaten sollen in einem gewünschten Intervall an eine beliebige E-Mail-Adresse gesendet werden. Die gesandten E-Mail-Nachrichten sollen aus dem Folder Outgoing Messages gelöscht werden.

4.2.4.2 Tätigkeit

Zum Versenden einer E-Mail wurde eine Klasse SendEMail angelegt. Die Klasse SendEMail beinhaltet die Methoden sendEMailMessages, Time und deleteEMail. An die Methode sendEMailMessages werden die Parameter String Content und String Header übergeben. Die gesendeten E-Mail-Nachrichten werden in den E-Mail-Folder SENT gespeichert. Es wird ein Message und ein Address Objekt erstellt. Das Address Objekt beinhaltet die E-Mail-Adresse des Empfängers. Das erzeugte Address Objekt wird an das Message Objekt übergeben. Weitere Parameter, die an das Message Objekt übergeben werden, sind der Header der Nachricht sowie der Message Content. Als Rückgabeparameter dient ein Boolescher Wert. Die Methode deleteEMail, löscht alle Nachrichten aus dem Folder INBOX, welche dem Übergabestring entsprechen. In der Methode deleteEMail wird ein Message Array erstellt. In diesem Array werden alle Nachrichten des Folder INBOX übergeben. Nach der Übergabe wird das Array durchsucht und mit dem übergebenen Header verglichen. Nach erfolgreicher Suche, wird die gefundene Nachricht aus dem INBOX Folder gelöscht. Aufgetretene Probleme und Lösungen:

4.2.4.2.1 Erste Problemstellung

Problem:

E-Mail-Nachrichten werden nicht gesendet.

Lösungsansatz:

Trotz Erzeugung eines Message Objektes und das Angeben eines Empfängers kann keine E-Mail gesendet werden. Laut Research in Motion Dokumentation muss ein Folder bei Erzeugung des Message Objektes angegebenen werden.

Lösung:

Laut der Dokumentation von Research in Motion, muss ein Folder bei der Erzeugung eines Message Objektes angegeben werden. Hierfür wird eine Instance store der Klasse Store erzeugt. Durch Session.getDefaultInstance().getStore(); wird diesem ein Store Objekt zugewiesen. Weiterhin wird ein Folder Array erzeugt. Diesem wird, mit Hilfe von store.list, eine Folder Liste zugewiesen. Die ausgehenden E-Mail-Nachrichten werden im Folder SENT abgelegt und versendet

4.2.4.2.2 Zweite Problemstellung

Problem:

Gesendete E-Mails können nicht gelöscht werden.

Lösungsansatz:

Wie in der oberen Problemstellung half mir hier die Dokumentation von Research in Motion.

Lösung:

Um E-Mail-Messages aus der INBOX löschen zu können, muss, wie im obigen Problem beschrieben, ein Array Folder Array erzeugt werden. Es wird jeder Header einer E-Mail-Message mit dem übergebenen String verglichen. Wird eine E-Mail-Message gefunden, wird diese aus dem Inbox-Folder gelöscht.

4.2.4.3 Qualitätskontrolle

In diesem Milestone wurde die Klasse Send-E-Mail erstellt, welche das Senden und Löschen von E-Mail-Messages bereitstellt. Es können E-Mail-Messages an eine E-Mail-Adresse gesendet werden. Nach Versand einer E-Mail-Message, wird diese aus dem INBOX-Folder gelöscht

4.2.4.4 Nächste Aufgabenstellung

In der nächsten Aufgabenstellung, soll eine API erstellt werden. Diese soll es dem Anwender ermöglichen das Senden von GPS-Positionsdaten zu beenden und zu starten und einen Arbeitsbeginn und ein Arbeitsende zu senden.

4.2.5 Milestone vier

4.2.5.1 Beschreibung

Milestone	Erstellen der Benutzeroberfläche
Zweck	Erstellen einer Benutzeroberfläche zum Starten und Stoppen des GPS-Listener
Abhängigkeiten	Die im Milestone zwei und drei entwickelten Programmteile gelten als Grundlage für diesen Projektteil.
Erforderliche Qualitätsprüfung	Die Benutzeroberfläche soll es den Anwender erlauben, den GPS-Listener zu starten und zu stoppen. Des weiteren soll dem Benutzer die Möglichkeit geboten werden, einen Arbeitsbeginn und ein Arbeitsende zu senden.

4.2.5.2 Tätigkeit

In diesem Projektteil wurde eine Benutzeroberfläche erstellt. Diese soll es dem Anwender ermöglichen, den Dienst zu starten und zu stoppen und einen Arbeitsbeginn und ein Arbeitsende zu senden. Um dies zu gewährleisten, wurde ein Hauptmenü erstellt. In dem Hauptmenü befinden sich drei Untermenüpunkte. In Abbildung 6.1 ist das Hauptmenü zu sehen. Die Untermenüpunkte sind

- Simpoint
- Clock IN/ Out
- About

Es wurde eine Klasse API erstellt, welche das Interface FieldChangeListener implementiert. Das Interface FieldChangeListener schreibt die Methode fieldChanged vor. Sobald ein Fieldelement angeklickt wird, wird diese Methode aufgerufen. Beim Programmstart wird ein Konstruktor aufgerufen, welcher das Hauptmenü durch den Aufruf der Methode makeMainMenu erstellt. Bei Auswahl und Klick eines Untermenüpunktes, wird die Methode fieldChanged aufgerufen. In diesem Menüpunkt wird zuerst das angeklickte Element ausgewertet. Es wird geprüft, um welches Element es sich handelt, und die entsprechende Methode makeMenuAbout, makeMenuClockINOUT und makeMenuSimpoint aufgerufen. In den Abbildungen 6.2 bis 6.4 sind die jeweiligen Untermenüpunkte zu sehen. In dem Menüpunkt Simpoint hat der Anwender die Möglichkeit, den Listener zu starten oder zu beenden. Bei Klick auf den Button Stop Positioning wird der LocationListener in der Klasse GPSListener auf null gesetzt. Nach diesem Vorgang wird der Button Stop Positioning entfernt und der

Button Start Positioning eingefügt. In dem Untermenü Simpoint wird ein Feld zum Eintragen der Jobreferenznummer und der Zeit erstellt. Die Nachricht wird gesandt, sobald der Anwender eine Jobreferenznummer eingetragen hat und den Button Clock IN oder Clock OUT klickt. Falls keine Jobreferenznummer eingegeben wurde, erscheint in dem Feld Status eine Warnmeldung, die den Anwender darauf hinweist, eine Jobreferenznummer einzugeben. Bei erfolgreichem Versand der Nachricht wird der Status auf send gesetzt. In dem letzten Menüpunkt about werden die Kontaktdaten des Unternehmens angeben.

4.2.5.2.1 Erste Problemstellung

Problem:

Die Methode fieldChanged wird nur von der „Return Taste“ aufgerufen. Das Trackwheel kann zum Klicken nicht verwendet werden.

Lösungsansatz:

In dem Blackberry-Forum erfuhr ich, dass vom Trackwheel die Methode navigationClick gefordert wird. Die Methode fieldChanged wird nur von Tastenelementen ausgelöst.

Lösung:

In der implementierten Methode navigationClick wird überprüft, welches ButtonField-Element angeklickt wurde. Wurde das Buttonfield-Element, welches den Event ausgelöst hat, identifiziert, wird die Methode fieldChanged aufgerufen und als Übergabeparameter das identifizierte Buttonfield-Element übergeben.

4.2.5.2.2 Zweite Problemstellung

Problem:

Die Applikation kann nicht in den Hintergrund gelegt werden.

Lösungsansatz:

In der Dokumentation erhielt ich keine Informationen über das Verstecken (hide) von Applikationen. Nur der Menüpunkt close Applikation wurde in der Dokumentation beschrieben. In einigen Foren erfuhr ich, dass diese Funktion für Java-Anwendungen nicht bereit stehe.

Lösung:

Nach langem recherchieren erfuhr ich, dass die Methode Application.requestBackground die laufende Applikation in den Hintergrund legt.

4.2.5.2.3 Dritte Problemstellung

Problem:

Zugriff von anderen Klassen auf EditField Elemente der Klasse API nicht möglich.

Lösungsansatz:

Mein Ziel war es, von der Klasse GPS-Listener aus, den Text des Feldes GpsStatusField zu verändern. Hierbei sollte der Status zwischen available (GPS-Daten vorhanden), unavailable (GPS nicht verfügbar) und Searching (suche nach GPS-Empfang) wechseln.

Lösung:

Mit dem Aufruf der Methode RedcoalAPI.getUiApplication(); erhielt ich eine Instance der Klasse UIApplication. Über diese Instance hatte ich Zugriff auf den aktiven Screen und konnte darüber das jeweilige zu verändernde EditField herauslesen.

4.2.5.3 Qualitätskontrolle

Die erstellte Applikation läuft stabil und wurde auf den Blackberry 8800 getestet. Leider steht uns momentan nur dieses Blackberry mit GPS-Funktion zur Verfügung, so dass ein Test auf anderen Geräten, nur auf dem Simulator möglich ist. Die Applikation sendet E-Mails in einem von dem Anwender gesetzten Intervall (Abbildung 6.5). Weiterhin wurde das Senden von Clock IN- und Clock OUT-Nachrichten getestet.

4.2.5.4 Nächste Aufgabenstellung

Als letzter Teil der Projektarbeit soll der E-Mail Client erstellt werden. Dieser soll auf einem Server laufen und in einem festgelegten Intervall E-Mails von einem E-Mail-Provider herunterladen und diese auswerten. Die ausgelesenen Koordinaten und Clock IN- und Clock OUT-Nachrichten sollen danach in einer Datenbank gespeichert werden.

5 Projektbeschreibung

In diesem Punkt soll die GPS-Anwendung für mobile Endgeräte beschrieben werden. Es werden die einzelnen Teilbereiche erläutert und ihre Funktionsweisen aufgezeigt. Dies soll Aufschluss darüber geben, wie die Anwendung entwickelt und aufgebaut wurde.

5.1 Überblick

Als erstes soll die Anwendung auf dem mobilen Endgerät erläutert werden. Es sollen Entscheidungsfindungen erläutert und deren Umsetzung beschrieben werden. Die Anwendung auf dem mobilen Endgerät ist für das Polling der GPS-Koordinaten zuständig. Diese werden in einem von dem Anwender wählbaren Zeitintervall empfangen und anschließend, nach einer Validierung, an einen Server gesendet. Unter Validierung der GPS-Positionsdaten ist hierbei die Prüfung zu verstehen, ob eine Positionsermittlung erfolgreich oder fehlerhaft verlief. Bei erfolgreicher Positionsermittlung werden die Positionsdaten zum Server gesendet. Bei fehlerhafter Positionsermittlung wird keine Positionsübertragung vorgenommen. Der zweite Teil dieser Projektbeschreibung gilt dem E-Mail Client. Dieser ist auf einem Server installiert und schaut in einem wählbaren Intervall nach eingegangenen E-Mails. Ist eine E-Mail eingegangen, wird diese vom E-Mail-Server heruntergeladen. Anschließend wird diese ausgewertet und die Positionsdaten in eine Datenbank gespeichert.

5.2 Vorbereitung

In der Vorbereitung wurde das Projekt spezifiziert. Es wurde ermittelt, welchen Umfang dieses Projekt haben soll. Auf Basis des geschätzten Projektumfangs wurden eine Projektdefinition und ein Anforderungskatalog erstellt. In der Projektdefinition wurden die zu erreichenden Ziele, der Umfang und der Projektansatz definiert. Desweiteren wurde in der Projektdefinition das Projektmanagementteam aufgestellt. Nachdem das Projekt definiert wurde, konnte die Qualitätserwartungen aufgestellt und eine Risikoabschätzung vorgenommen werden. Auf der Grundlage des Projektplans wurde das Projekt in Aufgabenbereiche unterteilt. Jeder Aufgabenbereich soll nachfolgend betrachtet werden.

Für den ersten Aufgabenbereich, der Erstellung der mobilen Anwendung, ist eine Programmstruktur definiert worden. Aufbauend auf der entwickelten Programmstruktur

wurde entschieden, welche Programmiersprache für die mobile Anwendung zum Einsatz kommen soll. Die zu verwendende Entwicklungsumgebung wurde anhand der zu nutzenden Programmiersprache festgelegt.

Als nächstes wurde der E-Mail Client definiert und ausgearbeitet. Zu Beginn wurde ein Projektteilplan aufgestellt. Anhand dieses Projektteilplans wurde der Projektumfang für diesen Teilbereich gesichtet und der Teilbereich anschließend in einzelne Aufgabenbereiche unterteilt

Nachdem die Anwendung in Teilbereiche gegliedert wurde, wurde eine Datenbankkonzeption entwickelt. Die Aufgabe bestand darin, bestehende Datenbanktabellen zu erweitern und neue Datenbanktabellen anzulegen.

Nachdem der Projektplan ausgearbeitet wurde und eine Struktur feststand, wurde eine zu verwendende Programmiersprache definiert. Aufbauend auf der zu verwendenden Programmiersprache wurde die Entwicklungsumgebung gewählt.

In dem nächsten Unterpunkt werden die eingesetzten Entwicklungsumgebungen aufgezeigt und beschrieben.

5.2.1 Mobile Anwendung

5.2.1.1 Jave Micro Edition (Java ME)

Auf Grundlage der definierten Anforderungen kommt die Programmiersprache Java zum Einsatz. Aufgrund der Plattformunabhängigkeit ist sie geeignet, eine Wiederverwendbarkeit des Codes, auch für andere mobile Geräte, zu garantieren. Dies vereinfacht das Portieren auf andere Plattformen erheblich, da nicht der komplette Code, sondern nur ein Teil des Codes angepasst werden muss.

5.2.1.2 Blackberry Entwicklungumgebung

Der Gerätehersteller bot uns eine Entwicklungsumgebung für das einzusetzende mobile Endgerät an. Wir entschieden uns, diese vom Hersteller angebotene Entwicklungsumgebung für die Umsetzung des Projektes zu verwenden.

5.2.2 Server Anwendung E-Mail-Client

5.2.2.1 Microsoft C#

Als Programmiersprache für den E-Mail-Client entschieden wir uns für die Programmiersprache C#.

5.2.2.2 Microsoft Visual Studio

Als Entwicklungsumgebung des E-Mail-Client wurde Microsoft Visual Studio gewählt.

5.2.2.3 Microsoft SQL Server

Der Microsoft SQL Server dient der Datenspeicherung der eingegangenen Informationen. Die Daten werden hierbei in bereits existierende oder neu angelegte Tabellen gespeichert.

5.3 Entwicklung

Die Gesamtprogrammierung wurde in zwei Aufgabenbereiche aufgeteilt. Der erste Aufgabenbereich betrifft die Umsetzung der mobilen Anwendung. In der zweiten Phase wurde der E-Mail-Client umgesetzt.

- Erstellen der mobilen Anwendung
- Umsetzen des E-Mail-Client

5.3.1 Erstellen der mobilen Anwendung

Dieser Aufgabenbereich wurde in vier Unteraufgaben aufgeteilt. Die erste Teilaufgabe beinhaltete die Installation und die Einarbeitung in die Entwicklungsumgebung. Dieser Aufgabenteil war deshalb wichtig, weil in den folgenden Aufgaben die volle Leistungsfähigkeit der Entwicklungsumgebung genutzt werden konnte. Im Folgenden werden die einzelnen Aufgabenbereiche aufgezeigt:

- Installation der Entwicklungsumgebung
- Erstellen des GPS-Listener
- Erstellen des Algorithmus zum Senden von Positionsdaten
- Entwicklung der Benutzeroberfläche

5.3.1.1 Installation der Entwicklungsumgebung

Es wurde bereits eine Woche vor Semesterbeginn mit dem Projekt begonnen. Hier wurden Informationen über die einzusetzende Software gesammelt. Die benötigte Software zum Erstellen der mobilen Anwendung wurde von der Webseite des Geräteherstellers Blackberry heruntergeladen. Zusätzlich zu der Entwicklungsumgebung Blackberry JDE 4.3.0 wurde der Blackberry Device Simulator

4.2.1.90 installiert. Dieser Simulator simuliert, das mir zur Verfügung stehende mobile Endgerät Blackberry 8800.

5.3.1.2 GPS-Listener

In dieser Teilaufgabe wurde eine Klasse zum Empfangen der GPS-Positionsdaten erstellt. Die Klasse GPSListener implementiert das Interface Thread. Da das Interface Thread implementiert wurde, musste die Methode nun überschrieben werden, die bei Start des Threads aufgerufen wird. Durch das Erstellen eines Konstruktor werden die Variablen Header, timeout, max_age und interval gesetzt, die für das Erzeugen des LocationListeners benötigt werden. Wird die Methode nun gestartet, wird ein Objekt von der Klasse LocationListenerImpl angelegt. Das erstellte LocationListenerImpl Objekt wird anschließend an das Objekt LocationProvider übergeben.

Nach dem Anlegen des LocationProvider läuft der Prozess in eine Endlosschleife. In der Endlosschleife wird die Statusvariable, welche die einzelnen Zustände angibt, ausgewertet. Die Statusvariable kann folgende Werte annehmen:

Der Wert der Statusvariable ist 0

Normalmodus: In diesem Fall werden die Positionsdaten in dem zuvor gewählten Intervall empfangen.

Der Wert der Statusvariable ist 1

Abbruch: Das Empfangen von Positionsdaten wird unterbrochen und der erstellte LocationProvider wird auf null gesetzt.

Der Wert der Statusvariable ist 2

Änderung: Die Intervallzeit zum Empfang von Positionsdaten wurde geändert. In diesem Fall wird ein neues LocationProvider Objekt erstellt und diesem die gewünschte Intervallzeit übergeben.

Befindet sich der Thread im Normalmodus, wird im ersten Schritt die aktuell ausgewählte E-Mail-Adresse des Anwenders ausgelesen. Nach diesem Schritt werden die positionsrelevanten Koordinaten von der Klasse LocationListenerImp*l* ausgelesen. Sind diese vorhanden, werden sie im nachfolgenden Schritt an die Klasse sendE-Mail übergeben. Nachdem die Nachricht an die Klasse sendE-Mail übergeben wurde, wird die Methode deleteE-Mail aufgerufen. Diese Methode löscht alle ausgegangenen positionsbezogenen E-Mail-Nachrichten. Ist das Löschen von E-Mail-Nachrichten abgeschlossen, wird überprüft, in welchem Status sich das GPS-Statusfeld der

Benutzeroberfläche befindet. Ist das GPS-Statusfeld null, dann wurden noch keine Positionsdaten von dem GPS-Empfänger empfangen und das GPS-Statusfeld wird auf Searching gesetzt. Nach Ende jedes Schleifendurchgangs, wird der Thread für zwei Sekunden in den Sleep-Modus gesetzt.

Innere Klasse LocationListenerImp

In der inneren Klasse LocationListenerImp wurde das Interface LocationListener implementiert. Durch das Implementieren des Interface LocationListener musste die Methode LocationUpdate bereitgestellt werden. Die Methode LocationUpdate stellt die Übergabeparameter LocationProvider und Location bereit. Ist die Location validiert, werden die Positionsdaten in einem Stringbuffer gespeichert und bereitgestellt. Nachfolgend wird der GPS-Status in der Benutzeroberfläche auf Available gesetzt. Falls die Positionsdaten nicht validiert wurden, wird der GPS-Status auf Unavailable gesetzt.

5.3.1.3 E-Mail-Versender

Die Klasse SendE-Mail besitzt die Methode sendE-MailMessage mit dem Übergabeparameter Content und Header und die Methode deleteE-M*ail* mit dem Übergabeparameter Header.

Bevor eine E-Mail gesendet wird, muss ein Folder angegeben werden, in dem die gesandte E-Mail gespeichert werden soll. Dieser Folder wird bei der Erstellung eines Message Objektes als Parameter übergeben. Ein *Message* Objekt besitzt die Methoden addRecipients, setSubjec*t* und setContent.

In der Methode addRecipients wird die Zieladresse eingetragen. Die Methode setSubject fügt das Subjekt in die E-Mail ein, welches den E-Mail-Kopf bildet.

An die Methode setContent werden die Positionsdaten und, falls gewählt, die Infomationen zum Arbeitsbeginn und Arbeitsende übermittelt. Zu diesen Daten werden des weiteren noch die identifikationsspezifischen Daten wie E-Mail und IMEI übergeben. Durch die Methode Transport.send wird die E-Mail gesendet.

Die Methode DeleteE-Mail löscht nach dem Versand die ausgegangene E-Mail aus dem Ausgangsordner. Das Löschen von Positions-E-Mails geschieht, da der E-Mail-Folder mit diesen Informationen überlaufen würde. Auch wäre es für den Nutzer unschön, wenn er hunderte Positionsnachrichten in seinem E-Mail-Ordner hätte. Beim Löschvorgang werden alle E-Mails im Ausgangsordner nach den übergebenen Header

durchsucht. Wird eine Nachricht mit übereinstimmendem Header gefunden, wird diese aus dem Ausgangsordner gelöscht.

5.3.1.4 Design und Menüaufbau der Benutzeroberfläche

In diesem Kapitel soll das Design und der Aufbau der API aufgezeigt werden. Die hier beschriebene API ist die Benutzeroberfläche der mobilen Anwendung. Für die Anwendung wurde ein Autostart-Event angelegt. Sobald das mobile Endgerät gestartet wird, läuft die Anwendung zum Empfangen und Senden von Positionsdaten. Die Anwendung, welche dem Nutzer zur Verfügung steht, wurde in drei Menüpunkte unterteilt.

- Simpoint
- Clock IN/ Out
- About

5.3.1.4.1 Hauptmenü

Öffnet der Benutzer die Anwendung, gelangt er in das Hauptmenü (Abbildung 6.1), in welchem ihm die drei, oben aufgeführten Menüpunkte zur Verfügung stehen. Das Statusfeld am Kopf der Anwendung zeigt, in welchem Modus sich der GPS-Empfänger befindet. Der GPS-Status kann zwischen “Searching”, “Available”, “Unavailable” und “Turn Simpoint On To Start” wechseln.

Ist der GPS-Status auf Searching gesetzt, sucht der GPS-Empfänger nach Satelliten. Dieser Status wird gesetzt, sobald noch keine Positionsermittlung stattfand.

Der Status „Available“ sagt aus, dass GPS-Positionsdaten empfangen wurden und validiert sind. Validiert heißt, dass Positionsdaten empfangen wurden und diese auswertbar sind.

Im Modus „Unavailable“ konnten die empfangenen GPS-Positionsdaten nicht validiert werden.

Im Status „Turn Simpoint On To Start” wurde „Simpoint“ gestoppt und es findet kein Empfang von GPS-Positionsdaten statt.

Der abgebildete Status hinter dem Button „Simpoint“ sagt aus, in welchen Zustand die Anwendung „Simpoint“ gesetzt wurde. Ist der Status auf „ON“ gesetzt, ist die Anwendung „Simpoint“ aktiv und es werden GPS-Positionsdaten empfangen und

gesendet. Ist der Status auf „OFF“ gesetzt, wurde die Anwendung gestoppt und es werden keine GPS-Positionsdaten empfangen und gesendet.

5.3.1.4.2 Menüpunkt „Simpoint“

Bei Wahl des Menüpunktes „Simpoint“ (Abbildung 6.2) hat der Anwender die Möglichkeit, das Empfangen und Senden von GPS-Positionsdaten zu stoppen und zu starten. Wird der zur Verfügung stehende Button „Stop Positioning “gedrückt, wechselt die Aufschrift des Button in „Start Positioning“.

Ist der Button mit der Aufschrift „Stop Positioning“ zu sehen, empfängt und sendet die Anwendung in dem vorgegebenen Intervall GPS-Positionsdaten. Bei der Aufschrift des Buttons „Start Positioning“ ist das Senden und Empfangen von GPS-Positionsdaten gestoppt. Der Prozess zum Senden und Empfangen von GPS-Positionsdaten kann gestartet werden, wenn man den Button erneut klickt.

Klickt der Anwender auf den Button „Done“, gelangt er in das Hauptmenü der Anwendung zurück.

5.3.1.4.3 Menüpunkt „Clock IN/ Out“

Der Menüpunkt „Clock IN/ Out“ (Abbildung 6.3) bietet dem Anwender die Möglichkeit einen Arbeitsbeginn und ein Arbeitsende zu senden. In der Abbildung 4.3 ist das Menü „Clock IN/ Out“ zu sehen. In dem Statusfeld im Menü „Clock IN/ Out“ wird der Sendestatus angezeigt. Um eine „Clock IN/ Out“-Nachricht senden zu können, muss eine Job-ID vergeben worden sein. Falls keine Job-ID vergeben wurde, wird in dem Statusfeld die Nachricht "Please Enter A Job Ref" angezeigt. Als „Job Reference“ sind sowohl Buchstaben als auch Zahlen zugelassen. In dem Feld „Enter Job Start/Stop Time“ kann die Zeit bei Job-Beginn eingetragen werden. Als Standardwert ist die aktuelle Zeit eingetragen. Nach dem Eintrag aller Daten kann der Nutzer eine Clock-Out für das Arbeitsende oder ein Clock-IN für den Arbeitsbeginn senden. Zusätzlich zu der Job-Reference und dem Datum werden die aktuellen Positionsdaten gesendet. Klickt der Benutzer den Button „Done“, gelangt er zum Hauptmenü zurück.

5.3.1.4.4 Menüpunkt „About“

In dem Menüpunkt „About“ (Abbildung 6.4) findet der Benutzer Kontaktdaten des Unternehmens. Klickt der Benutzer auf den Button „Back“, gelangt er in das Hauptmenü zurück.

5.3.1.4.5 Pop-Up-Menü

Wie bereits in dem obigen Kapitel 4.3.1.2 beschrieben, hat der Benutzer die Möglichkeit, die Intervallzeit des GPS-Listener einzustellen. Drückt der Benutzer den Blackberry-Button des mobilen Endgerätes, erscheint ein Menü, das es unter anderem erlaubt, die Intervallzeit zu setzen (Abbildung 6.5).

In Abbildung 6.5 ist das Pop-Up Menü zu sehen. Dem Anwender stehen hierbei fünf Intervallzeiten zur Verfügung. Hierbei können die GPS-Positionsdaten in den Intervallen von einer Minute, von fünf-, zehn- oder zwanzig Minuten bzw. jede Stunde empfangen und gesendet werden. Wird die Intervallzeit geändert, so empfängt und sendet der GPS-Listener die GPS-Positionsdaten in diesem gesetzten Zeitraum. Bei Auswahl des Pop-Up-Menüpunktes „Where Am I“ öffnet sich die von Blackberry bereitgestellte Anwendung „Blackberry-Maps“ und zeigt den aktuellen Aufenthaltsort des Benutzers auf einer Karte an. Bei Wahl des Menüpunktes „Hide“ verschwindet die Anwendung in den Hintergrund und läuft dort weiter und sendet und empfängt GPS-Positionsdaten.

Die Anwendung kann durch Drücken der Pfeil-Taste beendet werden. Ist die Anwendung beendet, werden keine GPS-Positionsdaten empfangen und gesendet.

5.3.1.5 Technische Umsetzung der Benutzeroberfläche

In diesem Kapitel 4.3.1.5 soll die programmiertechnische Umsetzung der Benutzeroberfläche näher erläutert werden. Es soll eine Übersicht der erstellten Klassen und Methode stattfinden. Aufbauend auf dieser Übersicht sollen der Aufbau der wichtigsten Klassen und Methoden und deren Verwendung beschrieben werden.

Die erstellten Klassen und Methoden sind in Abbildung 6.6 zu sehen.

Die Klasse RedcoalAPI:

Öffnet der Benutzer die Anwendung, wird die Main Methode aufgerufen. Diese erstellt ein Objekt der Klasse RedcoalAPI und ruft die Methode enterEventDispatcher auf.

Nachdem die Main Methode gestartet wurde, wird der Konstruktor der Klasse RedcoalAPI aufgerufen. In diesem Konstruktor wird die Klasse *API* durch die Methode pushscreen gestartet und auf den Bildschirm geladen.

In der Klasse API wird anschließend durch den Aufruf des Konstruktor die grafische Oberfläche geladen. Es wird das Bild gesetzt, der GPS-Listener gestartet und das Main Menu aufgebaut.

Die implementierte Methode Fieldchanged wird aufgerufen, sobald eine Taste des mobilen Endgerätes gedrückt wird. In dieser Methode werden alle Buttons der Anwendung behandelt. Wird ein Button gedrückt, wird dieser überprüft und entsprechend reagiert.

Die Methode CloseItem beendet die Anwendung bei Betätigung der Pfeiltaste.

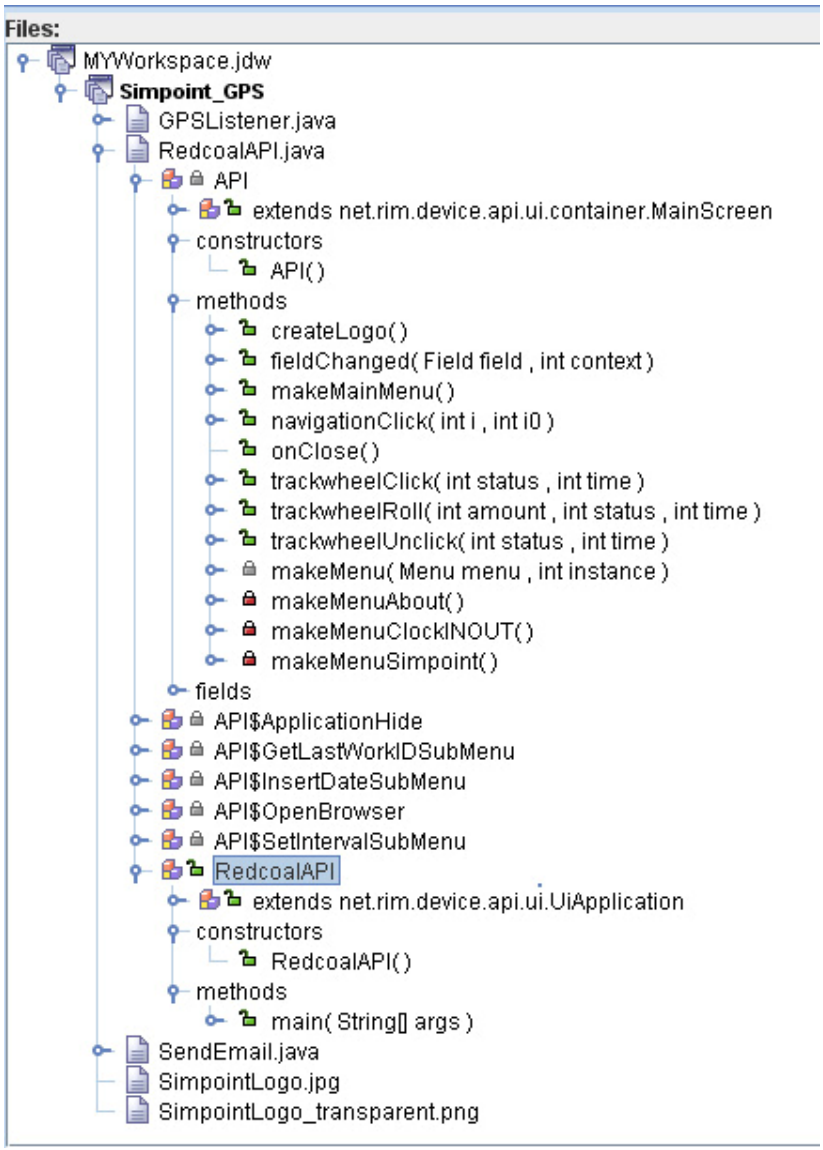

Abbildung 6.6 Klassen und Methodenübersicht mobile Anwendung

Die Methode MakeMenu erstellt das in Abbildung 4.4 dargestellte Pop-Up-Menü. Jedes SubMenu wird erstellt und in das Hauptmenü eingebunden. Die Intervallvergabe, der „Where Am I“- und der „Hide“-Menüpunkt sind in jedem Programmuntermenü zu sehen. In dem Menüpunkt Clock IN/ Out wird zusätzlich das SubMenu GetCurrentTime

eingefügt. Hier kann das aktuelle Datum und die aktuelle Zeit in das Feld „Job Start/Stop Time“ eingefügt werden.

In der Methode navigationClick wird die Aktion des TrackWheel ausgewertet. Sobald das Trackwheel gedrückt wird, wird nicht die Methode fieldchanged, sondern die Methode navigationClick aufgerufen. Hier wird ebenfalls geprüft, welcher Button gedrückt wurde und entsprechend darauf reagiert.

Die einzelnen Methoden MakeMenu, makeMenuSimpoin*t*, makeMenuClockINOUT und makeMenuAbout bauen die unterschiedlichen Programmmenüpunkt auf. Sobald ein anderer Menüpunkt aufgerufen wird oder der „Done- Button gedrückt wird, wird die Methode deleteRange aufgerufen, welche die momentan zu sehenden Felder wie Button- und Textfelder löscht.

Die einzelnen SubMenu-Klassen wurden angelegt, um auf die einzelnen Pop-Up-Menüpunkte zu reagieren.

5.3.2 Umsetzen des E-Mail-Client

In diesem Kapitel soll die Umsetzung des E-Mail-Client betrachtet werden. Der E-Mail-Client ist dafür verantwortlich, dass die von dem mobilen Endgerät gesendeten E-Mail-Positionsdaten von einem E-Mail-Server heruntergeladen und die Positionsdaten in eine Datenbank gespeichert werden. Der E-Mail-Client wurde zuerst in einem eigenständigen Programm entwickelt. Nachdem der E-Mail-Client getestet und die Funktionalität nachgewiesen wurde, konnte dieser in ein bestehendes Programm integriert werden. Dieses Programm wurde von mir in der Praxissemesterzeit entwickelt. Es liest eingegangene Positionsdaten eines GPS-Modems über UDP aus und speichert die eingegangenen Positionsdaten in eine Datenbank. Da diese beiden von mir entwickelten Programme GPS-Positionsdaten auslesen und speichern, wurden sie in einen Programm zusammengefasst, um die Anzahl der laufenden Programme auf den Server zu minimieren.

Der zu entwickelnde E-Mail-Client wurde in zwei Aufgabenbereiche unterteilt.

- Entwicklung des E-Mail-Clients
- Entwicklung der Stored Procedure

5.3.2.1 Entwicklung des E-Mail-Client

In der Abbildung 6.7 wird die Klassenstruktur des E-Mail-Client aufgezeigt. In den nachfolgenden Zeilen sollen die einzelnen Klassen beschrieben und deren Funktion erläutert werden. Wie bereits oben erwähnt, wurden in diesem Projekt zwei Listener implementiert. In dieser Projektbeschreibung soll lediglich der E-Mail-Client zum Auslesen der empfangenen E-Mails von einem E-Mail-Server beschrieben werden. Es wird zuerst die Klasse GPSListener beschrieben. Aufbauend darauf wird die Klasse Pop3MailClient zum Öffnen der Pop3-Verbindung und dem Herunterladen der E-Mails näher erläutert. Die Klasse Base64Decode*r* ist eine Fremdentwicklung. Sie wurde von der Seite http://www.codeproject.com/ heruntergeladen und in dieses Projekt implementiert. Zum Schluss werden die Klassen stored und Procedure beschrieben, welche zum Auslesen der Positionsdaten und zum Speichern dieser in die Datenbank verantwortlich sind. Die gesamte Anwendung ist eine Konsolenanwendung. Die zugrunde liegende Programmiersprache ist C#.

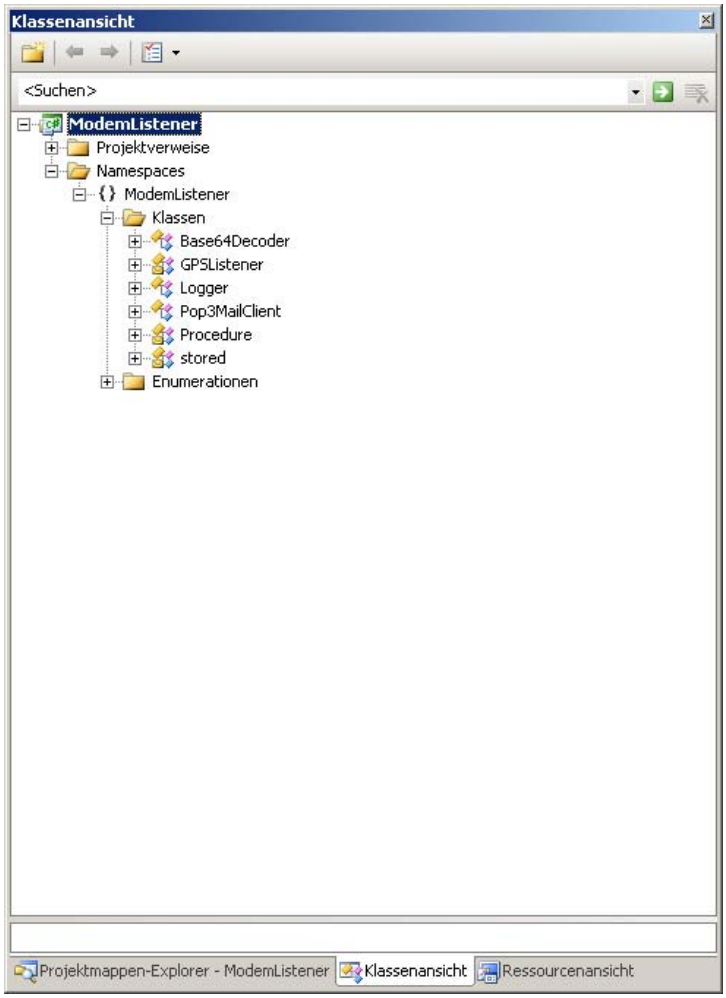

Abbildung 6.7 Klassen und Methodenübersicht E-Mail Client

5.3.2.1.1 Die Klasse GPSListener

Die Main Methode wird zu Beginn aufgerufen. In der Main Methode wird ein Objekt der Klasse GPSListener erstellt und die Methode StartListener aufgerufen.

In der Methode StartListener werden zwei Threads erzeugt. Der erste Thread startet den UDPClient. Der zweite Thread startet den hier zu betrachtenden E-Mail-Client. Bevor die beiden Threads gestartet werden, wird ein Timer gestartet. Dieser puscht jede Minute die aktuelle Zeit auf den Screen. Durch die Anzeige der aktuellen Zeit im Minutentakt kann festgestellt werden, ob die Anwendung noch läuft. In diesem Minutenintervall wird ebenfalls ein File gelöscht und wieder erzeugt. Der Filename enthält bei jedem Anlegen das aktuelle Datum und die aktuelle Uhrzeit. Von einer externen Anwendung wird dieses File auf Vorhandensein und Aktualität überprüft. Nachdem der Thread gestartet wurde, wird die Methode Pop3ListenerThread gestartet. Diese Methode erzeugt zu Beginn ein Objekt der Klasse Pop3MailClient. Als

Übergabeparameter werden hier der zu verwendende E-Mail-Server, der Port, der SSLMode, die E-MailAddress und das Password übergeben.

Nachdem das Pop3MailClient-Objekt angelegt wurde, läuft der Prozess in eine Endlosschleife. Jede Sekunde wird eine Verbindung zum E-Mail-Server aufgebaut. Dabei werden die aktuellen E-Mails heruntergeladen, die Daten ausgelesen und gespeichert und die Verbindung zum E-Mail-Server getrennt.

5.3.2.1.2 Die Klasse Pop3MailClient

In der Klasse Pop3MailClient werden durch ein Konstruktor die Variablen MailServer, Port , useSSL , Username und Password initialisiert.

In der Methode Connect wird ein neues TcpClient-Objekt angelegt. An diesem werden der Mail-Server und der Port übergeben. Durch eine angelegte Stream-Instanz wird ein neuer SslStream angelegt. Mithilfe der Methode AuthenticateAsClient wird die Authentifikation festgelegt. Durch Erstellen eines StreamReader-Objekts, an dem der SslStream übergeben wird, wird eine Verbindung mit dem E-Mail-Server aufgebaut. Nachdem eine Verbindung mit dem E-Mail-Server aufgebaut wurde, wird der Username verlangt. Durch den Befehl „USER username“ wird dem E-Mail-Server der Username übergeben. Als nächstes muss das Passwort eingegeben werden. Dieses wird mit dem Befehl „PASS password“ übergeben. Ist das Einloggen auf dem E-Mail-Server erfolgreich verlaufen, wird der Rückgabeparameter auf TRUE gesetzt. Ist ein Fehler in diesem Prozess aufgetreten, wird der Rückgabeparameter auf FALSE gesetzt.

Ist der Rückgabewert nach Aufruf der Connect Methode TRUE, wird die Methode storeCoordinates aufgerufen. Zu Beginn dieser Methode wird ermittelt, wie viele E-Mails auf dem E-Mail-Server vorhanden sind. Ist die Anzahl der E-Mails auf dem E-Mail-Server bekannt, wird eine While-Schleife durchlaufen und jede E-Mail durch den Befehl „RETR + Message ID“ zum Download angefordert. Der E-Mail-Server sendet nach Erhalt des Befehls „RETR“ die angeforderte E-Mail zeilenweise. Diese zeilenweise empfangene E-Mail wird in codierter Form von dem E-Mail-Server empfangen und muss nach Erhalt durch die Klasse Base64Decoder in ein Byte-Array decodiert werden. Dieses Byte-Array wird im nächsten Schritt mit einem ASCIIEncoding in ein ASCII-Zeichenformat decodiert. Ist die empfangene E-Mail komplett empfangen worden und in einem String gespeichert, wird diese formatiert. Um den String zu identifizieren, wird eine Zeichenkette „$Blackberry“ am Anfang des String eingefügt. Ist der String formatiert, wird die Methode storeDate der Klasse stored

aufgerufen. Hier werden die Positionsdaten ausgelesen und in die Datenbank gespeichert. Im Abschluss wird die E-Mail auf dem E-Mail-Server durch den Befehl „DELE + Message ID“ gelöscht.

Die Methode Disconnec*t* beendet die Verbindung zum E-Mail-Server durch den Befehl „QUIT“. Ist die Verbindung zum E-Mail-Server beendet, wird der Pop3Stream geschlossen.

5.3.2.1.3 Die Klasse Base64Decoder

Die Klasse Base64Decoder decodiert den vom E-Mail-Server empfangenen String in ein Byte Array. Diese Klasse wurde von der Seite http://www.codeproject.com/ heruntergeladen. Sie ist also nicht von mir erstellt worden. Die Klasse Base64Decoder erwartet als Übergabeparameter ein char-Array. Nach Aufruf der Methode getDecode wird das übergebene char-Array decodiert und ein Byte-Array zurückgegeben.

5.3.2.1.4 Die Klasse stored

In der Klasse strored wird der empfangene String ausgewertet und ausgelesen. Die ausgelesenen Positionsdaten werden anschließend in die Datenbank gespeichert. Der Methode start wird ein String übergeben. Dieser String enthält die Positionsinformationen. In der Methode start wird zu Anfang geprüft, um welches GPS-Gerät es sich handelt. Wie oben bereits erwähnt, empfängt diese Anwendung auch GPS-Positionsdaten von anderen GPS-Geräten. Diese Auswertung ist relevant, da die einzelnen GPS-Geräte die Positionsdaten an unterschiedlichen Stellen des Strings und in unterschiedlich codierter Form speichern. Ist der String ausgewertet und der eingegangene String als Blackberry String identifiziert, wird der String ausgewertet und ausgelesen. Es werden die Positionsdaten Longitude, Latitude, E-Mail, IMEI, Start Time, Finish Time, JobID und Clock ausgelesen. Ist der String erfolgreich ausgewertet und sind die Daten ausgelesen worden, wird die Variable StringIsValid auf 100 gesetzt. Eine If-Anweisung überprüft im nachfolgenden Schritt, ob die Variable StringIsValid auf 100 gesetzt ist. Ist dies der Fall, werden die ausgelesenen Positionsdaten in die Datenbank gespeichert.

Nach der Validierung des Übergabestrings wird geprüft, ob das Modem in der Datenbank vorhanden ist. Um diese Überprüfung durchzuführen, wird die Methode ModemExistCheck aufgerufen. In dieser Methode wird die Stored Procedure LBSGetIMEIFromMU2InputIdentifier aufgerufen. Als Übergabeparameter erhält diese Stored Procedure die Variablen E-Mail und IMEI Device. Der Rückgabeparameter ist

die IMEI Adresse des Gerätes auf der Datenbank. Für das Blackberry-Endgerät wird anhand der E-Mail-Adresse überprüft, ob dieses Gerät in der Datenbank vorhanden ist. Ist das Gerät in der Datenbank gespeichert und der Rückgabeparameter *IMEI DB* nicht null, wird dieser Wert, mit der IMEI Device als Schlüssel, in die Hash-Tabelle gespeichert. Der Vorteil dieser Hash-Tabelle ist, dass bei Empfang weiterer Positionsdaten nicht erneut eine Datenbankabfrage stattfinden muss. Es wird lediglich in der Hash-Tabelle nachgeschaut, ob die IMEI Device vorhanden ist und die zugehörige IMEI DB herausgelesen. Ist das Gerät vorhanden und die Variable IMEI DB nicht null, wird überprüft ob es sich bei der eingegangenen Nachricht um eine ClockIn/ClockOUT Nachricht handelt oder um Positionsdaten. Je nach eingegangener Nachricht wird die Stored Procedure GPSCreateJobEntry oder GPSCreateModemLocation aufgerufen.

5.3.2.1.5 Die Klasse Procedure

Die Klasse Procedure beinhaltet Methoden, die für den Aufruf der Stored Procedure verantwortlich sind. Um eine Stored Procedure auf der Datenbank aufzurufen, wird zu Beginn eine SqlConnection geöffnet. Die SqlConnection erwartet als Übergabe einen Connection-String. Dieser Connection-String enthält die IP des Datenbankservers, den Username, das Passwort und den Datenbanknamen. Als nächster Schritt wird ein SqlCommand erstellt. Diesem werden der Name der Stored Procedure und die erstellte SqlConnection übergeben. Anschließend wird durch SqlCommand.CommandType der Type Stored Procedure ausgewählt. Die Variablen werden mithilfe von SqlParameter übergeben. Hierbei unterscheidet man zwischen Input und Output Direction. Dies beschreibt, ob es sich um eine Eingabevariable oder Ausgabevariable handelt.

In der Abbildung 6.8 ist die laufende Server-Anwendung zu sehen. Sie zeigt, dass die ausgelesenen Koordinaten in die Datenbank gespeichert werden.

5.3.2.2 Umsetzen der Stored Procedure

In der Abbildung 6.9, 6.10 und 6.11 sind die erstellten Stored Procedure aufgezeigt. Diese laufen auf den Datenbankserver und können mithilfe der Methode callProcedure der Klasse Procedure aufgerufen werden. Die Strored Procedure in Abbildung 6.9 hat die Eingabeparameter E-Mail und IMEI. Als Ausgabeparameter werden die Variablen IMEI DB und MUID DB definiert. Aufgabe dieser Stored Procedure ist es, die IMEI-Adresse und die Mobile Unit ID des mobilen Endgerätes aus Datenbank zu lesen. Es wird hier unterschieden, ob es sich um ein Gerät vom Typ Blackberry oder um ein GPS-Modem handelt. Anhand der E-Mail-Adresse wird geprüft, ob das übergebene

Gerät eine Blackberry-Device ist. Anschließend wird anhand der E-Mail die IMEI_DB und die MUID DB aus der Datenbank gelesen. Existiert keine IMEI und MUIS auf der Datenbank zu dieser E-Mail, so handelt es sich um ein neues Gerät und die übergebene IMEI Device wird als IMEI eingefügt. Somit ist ein Gerät genau einer E-Mail zugeordnet.

In der Abbildung 6.10 wird ein Arbeitseintrag in die Datenbank eingefügt. In der Stored Procedure in Abbildung 6.11 wird eine neue Location in die Datenbank eingefügt.

5.4 Installation der Anwendung auf dem mobilen Endgerät

Um die Anwendung auf dem mobilen Endgerät zu installieren, muss diese von Research in Motion (RIM) signiert werden. Nachdem die Anwendung signiert wurde, kann diese mithilfe des Blackberry Javaloader auf das mobile Endgerät übertragen und installiert werden.

Der Befehl javaloader -usb load Anwendungsname.cod installiert die Anwendung auf dem mobilen Endgerät.

Der Befehl javaloader -usb erase -f Anwendungsname.cod deinstalliert die Anwendung auf dem mobilen Endgerät.

6 Anhang

Abbildung 6.1 – Main Menu

Abbildung 6.2 - Menu Simpoint

Abbildung 6.3 – Clock In und Clock Out Menu

Abbildung 6.4 – About Menu

Abbildung 6.5 – Applikation Menu

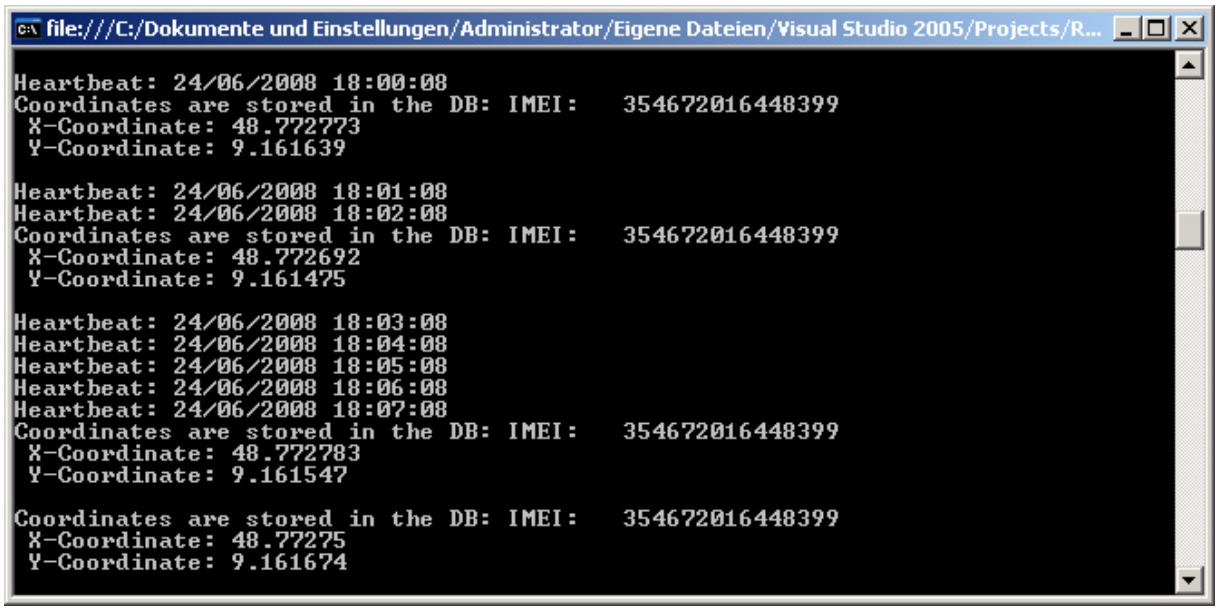

Abbildung 6.8 – Server-Anwendung

```
ALTER PROCEDURE [dbo].[ GetIMEI]
(
@E-MAIL nvarchar(50),
@IMEI_Device bigint,
@IMEI_DB bigint OUTPUT,
@MUID_DB int OUTPUT
)

AS
DECLARE @SearchWord varchar(30)
Declare @MobileUnitID int
SET @IMEI_DB = 0
BEGIN
IF @E-MAIL is not null
BEGIN
SELECT @IMEI_DB = IMEI, @MUID_DB = MobileUnitID
FROM MobileUnit2
WHERE E-Mailaddress = @E-MAIL

IF @IMEI_DB is null and @MUID_DB > 0
BEGIN
Update MobileUnit2 SET IMEI = @IMEI_Device WHERE E-Mailaddress = @E-MAIL
SET @IMEI_DB = @IMEI_Device
SET @MUID_DB = @MobileUnitID
end
END
END
RETURN
```

Abbildung 6.9–Stored Procedure GetIMEI

```
ALTER PROCEDURE [dbo].[CreateJobEntry](
@X float,
@Y float,
@IMEI bigint,
@GivenTime datetime,
@Type int,
@JobRef nvarchar(50)
)
AS

DECLARE @ID AS int
Declare @MobileUnitID int
Select @MobileUnitID = MobileUnitID
FROM MobileUnit2
WHERE IMEI = @IMEI

INSERT INTO Jobs(X, Y, GivenTime, MobileUnitID, JobRef,Type)
VALUES ( @X, @Y, @GivenTime, @MobileUnitID, @JobRef, @Type)

Select @ID = @@identity

RETURN scope_identity()
```

Abbildung 6.10 –Stored Procedure CreateJobEntry

```
ALTER PROCEDURE CreateModemLocation(
@X float,
@Y float,
@IMEI bigint
)
AS

DECLARE @ID AS int

INSERT INTO GPSLocations(X, Y, TimeStamp, IMEI)
VALUES ( @X, @Y, (getDate()), @IMEI)

Select @ID = @@identity

RETURN scope_identity()
```

Abbildung 6.11 –Stored Procedure CreateModemLocation