Diplomarbeit
Informatiker TS Uster

BluePhone

Erstellt am:
3. April 2004 bis 15. August 2004

Erstellt von:
Thomas Gemperli

Inhaltsverzeichnis

1 Übersicht 7
1.1 Übersicht 7
1.2 Rahmenbedinungen 7
1.3 Planung 7
1.4 Analyse 7
1.5 Design 7
1.6 Implementation 7
1.7 Tests 7
1.8 Bedienungsanleitung 8
1.9 Erfahrungsberichte und Erkenntnisse 8
1.10 Unterschriften 8
1.11 Anhang 8
2 Rahmenbedinungen 9
2.1 Aufgabenstellung der Schule/Betreuer 9
2.2 Bestimmungen der Technikerschule Uster 10
2.2.1 Zweck der Bestimmungen 10
2.2.2 Zulassung, Randbedingungen 10
2.2.3 Zielsetzungen 10
2.2.4 Themenvorschläge 10
2.2.5 Umfang der Arbeit 11
2.2.6 Dokumentation 11
2.2.7 Betreuer 12
2.2.8 Experte 12
2.2.9 Diplomanden 13
2.2.10 Präsentation der Arbeit 13
2.2.11 Bewertung der Arbeit 13
2.2.12 Rekurs 14
2.2.13 Eigentum der Arbeit 14
2.3 Bestimmungen der Me Myself and I AG 15
2.3.1 Allgemein 15
2.3.2 Dokumentation des Codes 15
2.3.3 Namesgebung der Elemente 16
2.3.4 Formatierung des Quellcodes 17
3 Planung 20
3.1 Pflichtenheft 20
3.1.1 Allgemeine Beschreibung 20
3.1.2 Einsatz 20
3.1.3 Software 20
3.1.4 Hardware 20
3.1.5 Funktionen 20
3.1.6 Schnittstellen 21

3.1.7 Leistungsanforderungen ... 21
3.1.8 Daten ... 22
3.2 Motivationsgründe ... 22
3.3 Aufwandschätzung ... 22
3.4 Projektplan ... 23
3.4.1 Projektplan Februar bis März ... 23
3.4.2 Projektplan April bis Mai ... 24
3.4.3 Projektplan Juni bis Juli ... 25
3.4.4 Projektplan August bis September ... 26
3.4.5 Vergleich „Soll – Ist“ der Projektplanung ... 26
3.5 Tatsächlicher getätigter Aufwand ... 27
4 Analyse ... 29
4.1 Vorabklärungen ... 29
4.1.1 Dokumentation zu Sony Ericsson Mobil Telephonen ... 29
4.1.2 Dokumentation zum GSM Standard ... 29
4.1.3 Dokumentation zu Bluetooth unter Linux ... 30
4.1.4 Dokumentation der seriellen Schnittstelle unter Linux ... 31
4.1.5 Dokumentation zu OpenOBEX ... 32
4.1.6 Kommunikation mit dem Mobiltelephon ... 33
4.2 Softwareentwicklung unter Linux ... 37
4.2.1 Übersicht ... 37
4.2.2 QT ... 37
4.2.3 QT Designer ... 37
4.2.4 Entwicklungsumgebung KDevelop ... 38
4.2.5 Doxygen ... 39
4.3 Systemgrenzen ... 40
4.3.1 Allgemein ... 40
4.3.2 Auflistung ... 40
4.4 Aktoren ... 40
4.4.1 Beschreibung der Aktoren ... 40
4.5 Use Case Diagramme ... 41
4.5.1 Use Case Diagramm erster Teil ... 41
4.5.2 Use Case Diagramm zweiter Teil ... 42
4.5.3 Use Case Diagramm dritter Teil ... 43
4.5.4 Use Case Diagramm vierter Teil ... 44
4.6 Beschreibung der Use Cases ... 45
5 Design ... 58
5.1 Entscheidungen ... 58
5.1.1 Programmiersprache ... 58
5.1.2 Bibliotheken ... 59
5.1.3 Entwicklungsumgebung ... 59
5.1.4 Datenhaltung ... 60
5.2 Klassendiagramm ... 61
5.2.1 Übersicht der Klassen ... 61
5.2.2 Definitives Klassendiagramm ... 62

5.3 Kollaborationsdiagramme 63
5.3.1 Allgemein 63
5.3.2 Kollaborationsdiagramm BluePhone 64
5.3.3 Kollaborationsdiagramm BlueLink 65
5.3.4 Kollaborationsdiagramm BlueConvert 66
5.3.5 Kollaborationsdiagramm BlueObex 67
5.3.6 Kollaborationsdiagramm BlueSettings 67
5.3.7 Kollaborationsdiagramm NewContact 68
5.3.8 Kollaborationsdiagramm NewSms 68
5.4 GUI 69
5.4.1 Allgemein 69
5.4.2 BluePhone “Phone” Tab 69
5.4.3 BluePhone “File” Tab 70
5.4.4 BluePhone “SMS” Tab 71
5.4.5 BluePhone “Contacts” Tab 72
5.4.6 BluePhone “Settings” Tab 73
5.4.7 BluePhone “Help” Tab 74
5.4.8 BluePhone “New SMS” Dialog 75
5.4.9 BluePhone “Reply SMS” Dialog 75
5.4.10 BluePhone “New Contact” Dialog 76
5.4.11 BluePhone “Edit Contact” Dialog 76

6 Implementation 77
6.1 Verwendete Komponenten 77
6.2 Spezielles 77
6.2.1 Status LEDs 77
6.2.2 OpenOBEX 78

7 Tests 79
7.1 Allgemeines 79
7.2 Modultests 79
7.3 Abschlusstests 79
7.4 Stabilitätstest 89
7.5 Auswertung der Testergebnisse 89

8 Bedienungsanleitung 90
8.1 Abhängigkeiten 90
8.2 Installation 90
8.3 Konfiguration 91
8.4 Dateien hochladen 91
8.5 SMS Dienste nutzen 92
8.6 Kontakte Dienste nutzen 92
8.7 Diverses 93

9 Erfahrungsberichte und Erkenntnisse 95
9.1 Negatives 95
9.2 Positives 95

9.3 Fazit 96
9.4 Ausblick 96

10 Unterschriften 97
10.1 Betreuer 97
10.2 Diplomand 97

11 Anhang 98
11.1 Begriffserklärungen 98
11.2 Quellenverzeichnis 99
11.2.1 Bücher 99
11.2.2 Links 99
11.3 Sitzungsprotokolle 102
11.3.1 Sitzungsprotokoll vom 23. März 2004 102
11.3.2 Sitzungsprotokoll vom 11. Mai 2004 103
11.3.3 Sitzungsprotokoll vom 30. Juli 2004 104
11.3.4 Sitzungsprotokoll vom 13. August 2004 105
11.4 CD 106
11.4.1 Inhaltsverzeichnis der CD 106

1 Übersicht

1.1 Übersicht

In dieser Übersicht wird der Inhalt aller Kapitel kurz beschrieben.

1.2 Rahmenbedinungen

Enthält die Rahmenbedinungen der Schule sowie die, die ich mir selber in Form der „My Myself and I AG“ gesetzt habe.

1.3 Planung

Dieses Kapitel enthält das Pflichtenheft des Projektes sowie meine meine Motivationen und Zielsetzungen.
Weiter wird die zeitliche Planung des Projektes dokumentiert.

1.4 Analyse

Dokumentiert die Analyse des Projektes, welche Ideen ich verfolgt habe. Es zeigt Problematiken und Lösungsvarianten. Es werden einige wichtige Punkte einzeln analysiert, um noch gezieltere Abklärungen zu treffen und um gravierende Probleme frühzeitig zu erkennen zu können.

1.5 Design

Dieses Kapitel geht auf das Design der Software ein, wie ich die Funktionen der Software in die Klassen und Methoden aufgeteilt habe. Es definiert wie die Implementation, basierend auf der Analyse, zu realisieren ist.

1.6 Implementation

Hier wird auf die Implementation eingegangen. Dies sowohl im Allgemeinen als auch auf einige spezielle Lösungen.

1.7 Tests

Dieses Kapitel dokumentiert die durchgeführten Tests.

1.8 Bedienungsanleitung

Beschreibt, wie das Programm installiert wird und wie man es bedient.

1.9 Erfahrungsberichte und Erkenntnisse

Dieses Kapitel zeigt einige persönliche Erkenntnisse und Erfahrungen, die ich aus der Durchführung dieses Projektes gezogen habe.
Weiter wird auf die Umsetzung des Projektes bezüglich des Pflichtenheftes eingegangen.

1.10 Unterschriften

Die Unterschriften des Entwicklers und des Betreuers.

1.11 Anhang

Im Anhang finden sich:

- die Begriffserklärungen
- das Quellenverzeichniss
- die Sitzungsprotokolle meines Betreuers

Weiter befindet sich am Ende dieses Dokumentes eine CD mit folgendem Inhalt:

- diese Dokumentation im PDF Format
- die Doxygen generierte Klassen-Dokumentation der Software
- der Quellcode der Software, bereit zum kompilieren
- die Installationspackete der Software im tar.gz, dpkg und rpm Format

2 Rahmenbedinungen

2.1 Aufgabenstellung der Schule/Betreuer

Sehr geehrter Herr Gemperli,

Sie erhalten die folgende Diplomarbeit zur selbstständigen Lösung zugeteilt:

„BluePhone“ Linux Tool for managing a Sony Ericsson mobile phone

Beschreibung:

Für die Betriebssysteme Mac OS X und Windows 2000 / XP existieren einige Applikationen, um über Bluetooth mit einem Sony Ericsson Mobile Phone zu kommunizieren. Dies beinhaltet unter anderem Adressbuch- und Kalender-Synchronisation, Datenaustausch und eine SMS Verwaltung.
Ziel dieser Diplomarbeit ist es, diese Funktionen auch unter dem Betriebssystem LINUX zur Verfügung zu stellen.

Anforderungen an diese LINUX-Applikation:

- Graphische benutzerfreundliche KDE Oberfläche
- Funktionen, die erfüllt werden müssen:
  - Kommunikation: Bluetooth
  - Datenaustausch: Bilder und Töne hochladen und herunterladen
  - Telefonbuch: Sichern, wiederherstellen und editieren
  - SMS: Lesen, sichern, verfassen und senden
  - Kalender: Sichern
- Mögliche Optionen:
  - Telefonbuch und Kalender mit KDE-Adressbuch und KDE-Kalender abgleichen und editieren
  - Kommunikation: Seriell, USB und IrDA

Aufgaben:

- Einarbeitung in die LINUX KDevelop Entwicklungsumgebung und KDE API
- Einarbeitung in Bluetooth-Protokoll Sony Ericsson und Abklärung der Realisierungsmöglichkeiten
- Verfassen eines Projektplanes und eines Pflichtenheftes
- Design und Implementation einer LINUX Applikation nach den Grundsätzen des Software-Engineering.
- Dokumentation aller Arbeitsschritte

Das Ziel ist ein lauffähiges System, welches demonstriert werden kann.

Austeilung der Aufgabenstellung:	Mittwoch,	7. April 2004
Abgabe der 2 Dokumentationen:	Dienstag,	17. August 2004

Freundliche Grüsse,

Technikerschule Uster

Kurt Burri, Betreuer

2.2 Bestimmungen der Technikerschule Uster

2.2.1 Zweck der Bestimmungen

Diese allgemeinen Bestimmungen regeln die Durchführung der Vordiplomarbeiten VDA und Diplomarbeiten DA an der Technikerschule Uster.

Die Abläufe sind im Qualitätsmanagement System QMS beschrieben unter www.iso.gotdns.com/domasy/htm und erfolgen gemäss jährlich neu erstelltem Terminplan.

2.2.2 Zulassung, Randbedingungen

Zur Vordiplomarbeit wird zugelassen wer im 4. Semester studiert.
Zur Diplomarbeit wird zugelassen wer im Abschlusssemester studiert.
Die Arbeit kann innerhalb von zwei Jahren einmal wiederholt werden.

2.2.3 Zielsetzungen

Allgemeine Ziele

Der Diplomand
- plant und bearbeitet das Thema als Projekt. (Anwendung Projektmanagement)
- führt die Arbeit methodisch und strukturiert durch.
- bearbeitet das Thema selbständig oder im Team (maximal drei Mitglieder)
- dokumentiert die Arbeit strukturiert, verständlich, vollständig und nachvollziehbar
- präsentiert die wesentlichen Ergebnisse der Arbeit in kurzer und prägnanter Form.

Vordiplomarbeiten (TS)

Der Diplomand vertieft Gelerntes und wendet es fächerübergreifend an. (Vernetzung)

Diplomarbeit (TS und NDS)

Der Diplomand vertieft Gelerntes und wendet es fächerübergreifend an. (Vernetzung)
Er wählt das Thema so, dass in einem bestimmten Bereich Neues dazugelernt wird (Selbständiges Aneignen von Wissen).

2.2.4 Themenvorschläge

Die Vorschläge sollen bevorzugt von der Arbeitgeberfirma oder aus dem persönlichen Interessengebiet stammen.

Die Arbeit sollte möglichst ein in sich abgeschlossenes Thema sein.

Das Thema muss so gewählt werden, dass in einem bestimmten Bereich für den Diplomanden eine Herausforderung besteht.

Das Thema soll so gewählt werden, dass nach Möglichkeit eine Realisierung gemacht werden kann.

Bei der Themenwahl soll die Komplexität und der Aufwand berücksichtigt werden. Umfangreiche Arbeiten sollen im Team durchgeführt werden.

Bei der Themenwahl soll die benötigte Infrastruktur mit berücksichtigt werden.

Die Arbeit soll anlässlich der Präsentation im Schulhaus funktionell vorgeführt werden können.

2.2.5 Umfang der Arbeit

	Vordiplomarbeit	Diplomarbeit TS	Diplomarbeit NDS
Aufwand	150 Stunden	200 Stunden	100 Stunden
Seitenzahl Dokumentation ohne Anhang	25 Seiten	60 Seiten	25 Seiten

Die Angaben sind Richtgrössen bei Einzelarbeiten oder pro Teammitglied.

Die Arbeitsaufwendungen gemäss obigen Richtgrössen müssen in der Freizeit erbracht werden.

2.2.6 Dokumentation

Die Dokumentation soll ein getreues Abbild der ganzen Arbeit darstellen und folgende Struktur aufweisen:

Planung
Inhaltsverzeichnis, Aufgabenstellung, Glossar, Zusammenfassung der Ergebnisse, priorisierte Anforderungsspezifikationen, Projektplan

Engineering
Erarbeitung der Grundlagen, Vorabklärung und Analyse, Lösungsvarianten mit Vor- und Nachteilen und begründetem Entscheid der gewählten Variante.

Realisierung
Schrittweise Realisierung der gewählten Variante mit Einzeltests, Gesamttests mit Validierung der Funktionsanforderungen, Installations- und Benutzeranleitung, Verbesserungsvorschläge, persönlicher Rückblick mit Erkenntnissen.

Anhang
Quellenverzeichnis, Bestückungspläne, eventuell Stücklisten, Konstruktionszeichnungen, Datenblätter und Beschreibungen von Fremdprodukten, Geräten, Baugruppen und speziellen Bauelementen, Programmlistings.

Form
Die Dokumentation muss einseitig bedruckt im Format A4 sein und mit Plastikringen gebunden werden.
Auf einer beschrifteten CD befindet sich der Anhang, die Software als Quell- wie auch ausführbarer Code und die Dokumentation (Word, pdf). Diese wird hinten mit einer Papierhülle in die Dokumentation eingebunden.

2.2.7 Betreuer

Anforderungen
Der Betreuer ist vorzugsweise ein Fachlehrer der TSU. Es kann auch ein externer Betreuer sein, sofern dieser über einer höhere Ausbildung (TS, HF, HTL, FH, ETH, HWV, Uni) verfügt und keine kollegiale, oder verwandtschaftliche Beziehung zum Studierenden besteht.

Aufgaben
Der Betreuer formuliert die Aufgabenstellung gemäss Auftrag der Schulleitung. Er unterstützt den Diplomanden primär in der Projektabwicklung und nicht in fachlichen Fragen.
Des Weiteren fördert er das selbständige Arbeiten des Diplomanden, überwacht den zeitlichen Ablauf und fällt allenfalls notwendige Abweichungsentscheide von der Aufgabenstellung.
Während der Ausführungsphase führt er auf Verlangen des Studierenden mindestens folgende Besprechungen durch:

	Vordiplomarbeit	Diplomarbeit TS	Diplomarbeit NDS
Anzahl Besprechungen	2	4	2

Jede Besprechung wird durch den Betreuer protokolliert und beidseitig visiert.

Nach der Abgabe korrigiert und bewertet der Betreuer die Arbeit eingehend anhand der Dokumentation mit Hilfe des Formulars “Beurteilungshilfsblatt Diplomarbeit”.

Der Betreuer nimmt zusammen mit dem Experten an der Präsentation der Arbeit durch den Diplomanden teil und legt die Noten mit dem Experten fest.

2.2.8 Experte

Anforderungen
Der Experte ist kein Fachlehrer der TSU. Er verfügt über eine höhere Ausbildung (HTL, FH, ETH, HWV, Uni). Er übernimmt nur Bewertungen von Arbeiten die seinen Fachkenntnissen entsprechen. Es darf keine geschäftliche, keine kollegiale und keine verwandtschaftliche Beziehung zum Studierenden bestehen.

Aufgaben

Der Experte validiert bei den Diplomarbeiten TS die durch den Betreuer formulierte Aufgabenstellung.
Er beurteilt die Arbeit anhand der Dokumentation in unabhängiger Weise und kontrolliert die Bewertung des Betreuers.
Der Experte nimmt zusammen mit dem Betreuer an der Präsentation der Arbeit durch den Diplomanden teil und legt die Noten mit dem Betreuer fest. Der Experte moderiert bei den Vordiplomarbeiten und Diplomarbeiten TS den Präsentationsablauf. Ein Experte übernimmt pro Mal mindestens zwei bis maximal sechs Arbeiten.

2.2.9 Diplomanden

Er nimmt folgende Aufgaben gemäss Terminplan wahr:

- Er reicht ein bis zwei Themenvorschläge ein mit Hilfe des Formulars „Vorschlag für Diplom- / Vordiplomarbeit“
- Er vereinbart den ersten Besprechungstermin mit dem Betreuer.
- Er gibt zwei Dokumentationen für die Bewertung ab.
- Er präsentiert die Arbeit gemäss Präsentationsplan.

Bei Nichteinhalten der Termine schliesst die Schulleitung den Diplomanden aus der laufenden Diplomarbeitsperiode aus. Nach einem Ausschluss oder nach einem Abbruch durch den Diplomanden, kann er die Arbeit gemäss Promotionsbestimmungen an der folgenden Diplomarbeitsperiode mit einem anderen Thema einmal wiederholen.

2.2.10 Präsentation der Arbeit

Die Präsentation soll ein Vortrag über die wesentlichen Ergebnisse und eine praktische Vorführung der Arbeit sein. Die Vorstellung findet gemäss Terminplan statt, der nach Abgabe der Dokumentationen durch die Schulleitung erstellt und veröffentlicht wird.

Die Vordiplom- oder Diplomarbeit TS wird während 10 Minuten dem Experten und dem Betreuer vorgestellt. Während weiteren 10 Minuten stellt der Experte Fragen zur eigentlichen Arbeit und zum weiteren Umfeld. Bei Teamarbeiten verlängern sich die Zeiten proportional.

Die Diplomarbeit NDS wird während 15 Minuten der ganzen Klasse vorgestellt. Experten und Betreuer sind anwesend. Sie und die ganze Klasse stellen anschliessend während 10 Minuten Fragen zur Arbeit. Bei Teamarbeiten verlängern sich die Zeiten proportional.

2.2.11 Bewertung der Arbeit

Betreuer und Experte bewerten die Arbeit unabhängig anhand der Dokumentation. Als Kriterien und Hilfe verwenden sie dazu das Formular „Beurteilungshilfsblatt Vordiplom- oder Diplomarbeiten“.

Nach der Präsentation besprechen sie die Bewertung und erteilen für jedes Hauptkriterium eine Zehntelsnote. Die Berechnung der Gesamtnote erfolgt mit dem Formular „Bewertung Vordiplom- oder Diplomarbeit“.

Die Expertenkommission vergleicht innerhalb einer Woche die Notengebung untereinander um eine möglichst ausgeglichene Notengebung sicherzustellen. Ergeben sich Änderungsvorschläge, so werden diese mit dem betreffenden Experten und Betreuer diskutiert.

Anschliessend wird die Bewertung dem Diplomanden durch Zustellung des unterschriebenen Formulars „ Bewertung Vordiplom- oder Diplomarbeit“ bekannt gegeben. Falls der Diplomand detailliertere Auskünfte über die Bewertung möchte, kann er diese beim Betreuer nachfragen.

Teamarbeiten werden solidarisch bewertet, wenn eine gemeinsame Dokumentation abgegeben wird.

2.2.12 Rekurs

Lassen sich Unstimmigkeiten oder Streitigkeiten in der Bewertung nicht direkt oder mit Hilfe der Schulleitung lösen, kann der Diplomand einen Rekurs machen. Dieser wird gemäss Prozessbeschreibung „5.2 Rekurs behandeln“ abgewickelt.

2.2.13 Eigentum der Arbeit

Die Arbeit und das erworbene Know-how gehören dem Diplomanden und je nach Beteiligung, eventuell auch dem Auftraggeber.
Eine Dokumentation bleibt bei der Schule und kann weiteren Studierenden zu Studienzwecken zugänglich gemacht werden. Auf Verlangen verpflichtet sich die Schule jedoch zur Geheimhaltung.
Besteht von dritten ein kommerzielles Interesse an einer Arbeit, so leitet die Schule dieses an den Eigentümer der Arbeit weiter.

2.3 Bestimmungen der Me Myself and I AG

2.3.1 Allgemein

Die Me Myself and I AG repräsentiert den Auftraggeber dieser Arbeit. Da ich selber mein Auftraggeber für diese Diplomarbeit bin, dient dieser virtuelle Auftraggeber, soweit möglich, der Realitätsnähe dieses Projektes und der dazugehörigen Dokumentation.

2.3.2 Dokumentation des Codes

Der Quellcode muss, den Programmierrichtlinien des Betriebes entsprechend, im „Doxygen“ Stil dokumentiert werden.
Die API Dokumentation des Source Codes kann nach einer Änderung automatisch mittels Doxygen aktualisiert werden.
Eine detailierte Beschreibung der Doxygen Dokumentations Richtlinien findet sich unter http://www.doxygen.org

Im Wesentlichen sind zwei Arten von Dokumentations Blöcken zu unterscheiden:

Öffentliche Blöcke

Sämtliche Klassen, alle Methoden und alle in den Headerfiles (.h) definierten Variablen, auch private, müssen Doxygen parsebar dokumentiert sein.

```
/**
  * Dieser Dokumentationsblock wird von Doxygen geparst.
  */

/** Dieser ebenfalls. */
```

Private Blöcke

In den Quelldateien (.cpp) sollen Details einer Methode innerhalb dieser in konzentrierter Form und für Doxgen nicht parsebar dokumentiert werden.

```
/*
 * Dieser Dokumentationsblock wird von Doxygen nicht geparst.
 */

/* Dieser ebenfalls nicht. */
```

Die Dokumentationsblöcke, auch einzeilige, werden grundsätzlich mit /* geöffnet und mit */ geschlossen. Die Sterne zu Beginn jeder Zeile sind nicht zwingend.

```
/**
    Auch dieser oeffentliche Dokumentationsblock entspricht den Kriterien.
*/


/* Dieser private ebenfalls. */


// Dieser Stil der Dokumentation ist nicht erwuenscht.
```

Alle Files des Sourcecodes müssen mit einem dem folgenden Beispiel entsprechenden Header begonnen werden.

```
/**
    @class    MyClass
    @brief    Eine kompakte Beschreibung der Klasse.
    @author   Hans Müller, hans@domain.com
    @version  0.1
    @date     2004-06-11
    @par      This program is free software; you can redistribute it
              and/or modify it under the terms of the
              GNU General Public License.
    @file     myclass.h oder myclass.cpp
*/
```

Spezielles:

Da der Endkunde eine in Deutsch unterrichtende Schule ist, soll die gesamte Dokumentation in deutscher Sprache abgefasst werden.

Trotzdem sind Umlaute in den Dokumentationsblöcken des Sourcecodes tabu.

2.3.3 Namesgebung der Elemente

Allgemein

Klassen, Methoden und Variablen müssen mit sprechenden Namen und einer logischen Struktur folgend benannt werden.

Mehrere Bestandteile eines Namens werden mit Grossbuchstaben unterschieden.

Alle Namen werden in englischer Sprache geschrieben.

Variablen

Variablennamen beginnen immer mit Kleinbuchstaben, privaten Variablen muss der Prefix m_ vorgestellt werden.

```
int exampleLoopCounter;

QString m_exampleString;
```

Methoden

Methodennamen beginnen mit Kleinbuchstaben. Die Funktion der Methode soll aus dem Namen dieser ersichtlich sein. Dies bedingt meistens ein Verb zu Beginn des Methodennamens.

```
int  getSomeValue();

void setSomeValue( int someValue );

void connectToDevive();
```

Slots und Signale

Namen von Slots und Signalen beginnen mit Kleinbuchstaben. Die Funktion des Slots respektive des Signals soll aus dem Namen dieses ersichtlich sein. Anders als Methoden beginnen Slots mit dem Namen des Widgets, das sie aufruft. Dies ist im allgemeinen kein Verb.

```
virtual void settingsSave();

virtual void fileSend();
```

Klassen

Klassennamen beginnen mit einem Grossbuchstaben.

```
class MyClass : public OtherClass
{
}
```

2.3.4 Formatierung des Quellcodes

Headerfiles besitzen die Endung .h
Sourcefiles besitzen die Endung .cpp

Blöcke werden mit dem Tabulator eingerückt.

Geschweifte Klammern {} werden vertikal auf einer Linie ausgerichtet.

Der Inhalt von Kontrollstrukturen muss in Klammern gefasst werden, auch wenn der Compiler dies nicht ausdrücklich verlangt.

Ein Beispiel Header File:

```
/**
    @class    MyClass
    @brief    Eine kompakte Beschreibung der Klasse.
    @author   Hans Müller, hans@domain.com
    @version  0.1
    @date     2004-06-11
    @par      Kurzer Lizenztext
    @file     myclass.h
*/


#ifndef MYCLASS_H
#define MYCLASS_H


#include "myclassdlg.h"


/**
  * Eine ausfuehrliche Beschreibung der Klasse die sich auch
  * ueber mehrere Zeilen hinziehen kann.
  */
class MyClass : public OtherClass
{
    Q_OBJECT

public:

    /** Eine Beschreibung des Konstruktors. */
    MyClass();

    /** Eine Beschreibung des Destruktors. */
    ~MyClass ();

    /** Eine Beschreibung der Methode. */
    int  getSomeValue();


public slots:

    /** Eine Beschreibung des Slots. */
    virtual void settingsSave();


private:

    /** Eine Beschreibung der Variable. */
    int m_someResult;

};

#endif
```

Ein Beispiel Source File:

```
/**
    @class    MyClass
    @brief    Eine kompakte Beschreibung der Klasse.
    @author   Hans Müller, hans@domain.com
    @version  0.1
    @date     2004-06-11
    @par      Kurzer Lizenztext
    @file     myclass.cpp
*/

#include "myclass.h"


/** Eine Beschreibung des Konstruktors. */
MyClass::MyClass();

/** Eine Beschreibung des Destruktors. */
MyClass::~MyClass();


/**
  * Eine Beschreibung der Methode.
  */
int MyClass::getSomeValue()
{
    /* Interner Kommentar zum folgenden Code */
    if ( something )
    {
        return m_someResult;
    }
    else
    {
        m_someResult = something;
        return m_someResult;
    }
}
```

3 Planung

3.1 Pflichtenheft

3.1.1 Allgemeine Beschreibung

Es soll eine benutzerfreundliche Applikation erstellt werden, um mittels einem PC mit einem Sony Ericsson Mobiltelephon zu kommunizieren.

3.1.2 Einsatz

Die Software soll auf einem PC kompatiblen Rechner unter dem Betriebssystem Linux eingesetzt werden. Von Linux unterstützte und betriebsbereite Bluetooth Hardware am Rechner wird vorausgesetzt.

3.1.3 Software

Das GUI soll Benutzerfreundlich realisiert werden. Die benötigten Bibliotheken sind heute auf jeder gängigen GNU/Linux Distribution enthalten.

Die Software soll so gegliedert sein, dass eine Weiterverwendung des GUI's und dessen zugrunde liegenden Funktionen auf anderen, die Graphikbibliothek unterstützenden Betriebssystemen, möglich ist. Konkret bedeutet dies, dass hardwarespezifische Funtionen in Klassen gekapselt sein sollen.

Das GUI der Applikation soll in englischer Sprache gehalten sein.

3.1.4 Hardware

Die Software soll kompatibel sein zum Sony Ericsson T610 Mobiltelephon.

Falls alle Modelle der T-Serie ihrer Dokumentation entsprechend implemtiert sind, umfasst dies automatisch auch die folgenden Modelle: T68, T610, T630, beschränkt (nur SMS) auch die älteren T39 und T300.

Die Me Myself and I AG stellt die Modelle T68, T610 und T630 zu Testzwcken zu Verfügung.

Als Entwicklungsgeräte dienen ein T610 Mobiltelephon, ein Dell Inspiron 8000 PC Laptop und handelsübliche Bluetooth Dongle des Herstellers Acer.

3.1.5 Funktionen

Funktionale muss Anforderungen an die Applikation:

Bedienung
- Benutzerfreundlich und leicht zu erlernen

Kommunikation
- Bluetooth

Datenaustausch
- hochladen Bilder
- hochladen Töne
- herunterladen Bilder
- herunterladen Töne

Telephonbuch
- sichern
- wiederherstellen
- editieren

SMS
- lesen
- sichern
- vefassen & senden

Kalender
- sichern

Funktionale kann Anforderungen an die Applikation:

Kommunikation
- Serielles Kabel
- USB Kabel
- IrDA

Telephonbuch
- abgleichen mit KDE-Addressbook

Kalender
- abgleichen mit KDE-Calendar

3.1.6 Schnittstellen

Die Applikation soll Bluetooth als Schnittstelle zum Mobiltelephon verwenden. Optional kann die Software auch IrDA oder ein serielles Kabel als Schnittstelle unterstützen.

3.1.7 Leistungsanforderungen

Es sind keine speziellen Leistungsanforderungen an die Applikation gesetzt, da sich die Leistung eines Mobiltelefon in keinem Fall der Leistung eines PC's annähern wird.

Grundsätzlich soll die Applikation nicht mehr Ressourcen benötigen als eine ähnlich umfangreiche KDE Applikationen.

3.1.8 Daten

Gewonnene Daten sollen, um maximale Portabilität, Weiter- und Wiederverwendbarkeit dieser sicherzustellen, als reine Textdateien auf persistente Datenträger geschrieben werden.

3.2 Motivationsgründe

Für die Betriebsysteme Mac OS X und Windows 2000 / XP existieren einige Applikationen, um über Bluetooth mit einem Mobil Telephon zu kommunizieren. Dies beinhaltet u.A. Addressbuch und Kalender Synchronisation, Datenaustausch oder eine SMS Verwaltung.
Diese Anwendungen fehlen für das Betriebssystem Linux, das ich primär verwende. Ich möchte den Funktionsumfang meines Bluetooth fähigen Mobiltelephons jedoch auch unter Linux nutzen können.
Die Implementation einer eigenen Applikation ist sicher eine gute Möglichkeit, meine, in der Schule theoretisch erworbenen, Programmierkenntnisse in der Praxis einzusetzen und zu vertiefen.
Weiter möchte ich der Open Source Community etwas zurückgeben, die mir die Software, mit der ich täglich arbeite, unentgeltlich zur Verfügung stellt.

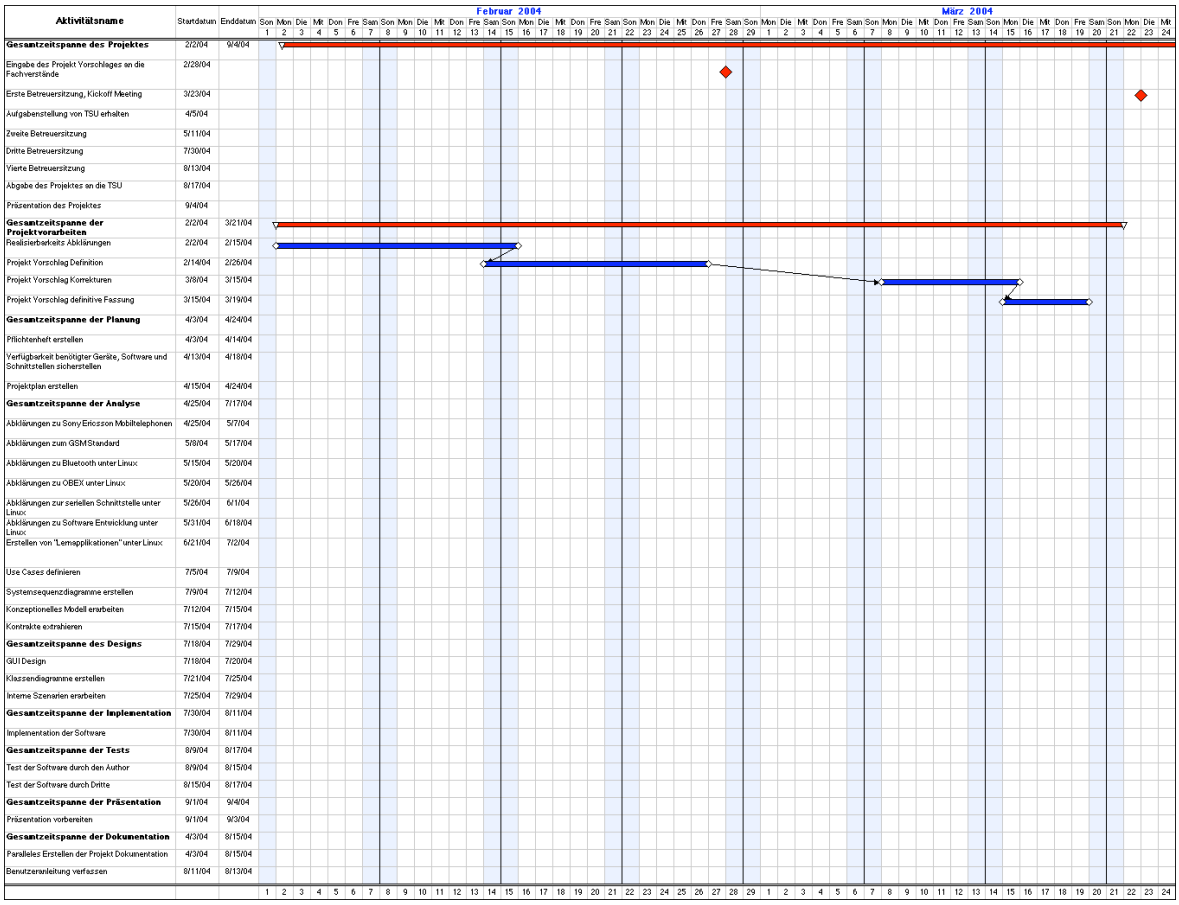
3.3 Aufwandschätzung

Diese Schätzung wurde zwecks Einreichung des Projektes an die TSU im Februar 2004 erstellt.

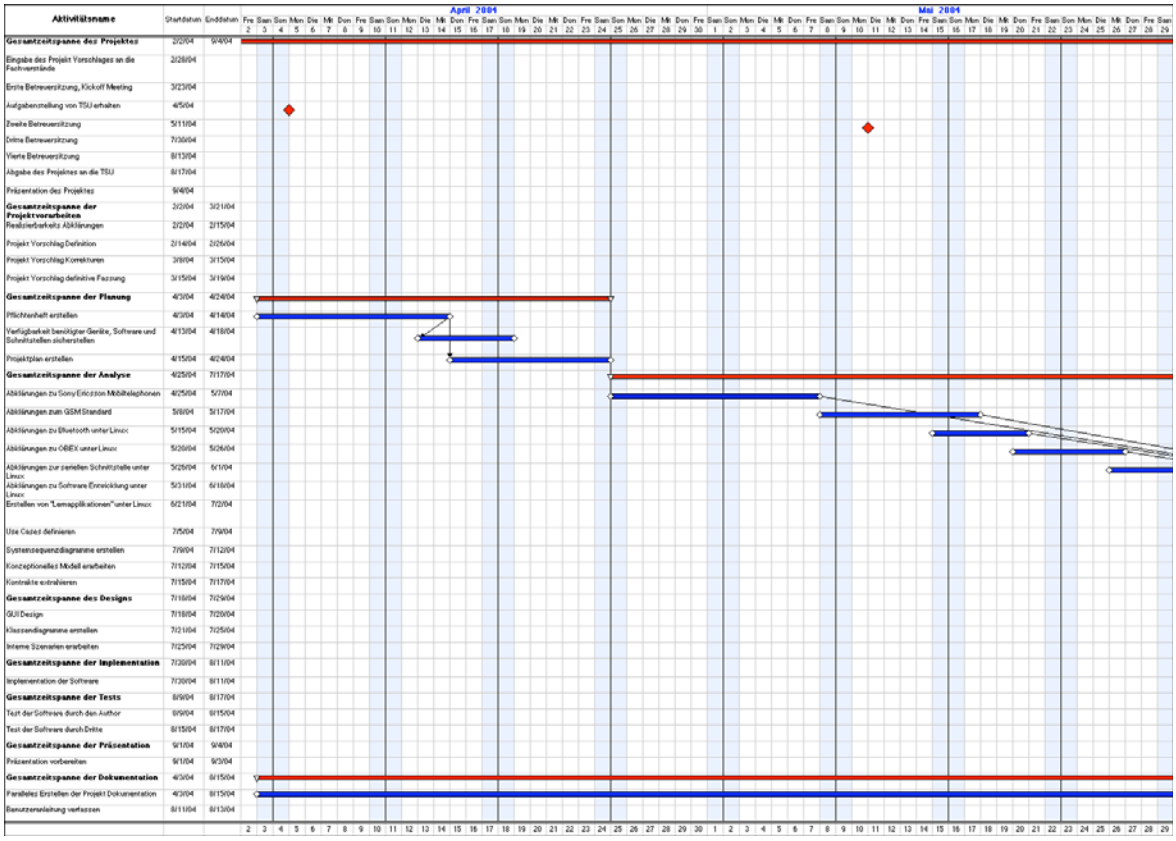
Aufgabe	Aufwand in Stunden
Vorabklärungen	15
Betreuersitzungen	10
Pflichtenheft erstellen	5
Analyse	30
Design	25
Implementation	30
Dokumentation	50
Tests	15
Korrekturen Software	10
Korrekturen Dokumentation	5
Bedienungsanleitung	5
Unvorhergesehenes	10
Präsentation vorbereiten	5
Benötigte Gesamtzeit	**215**

3.4 Projektplan

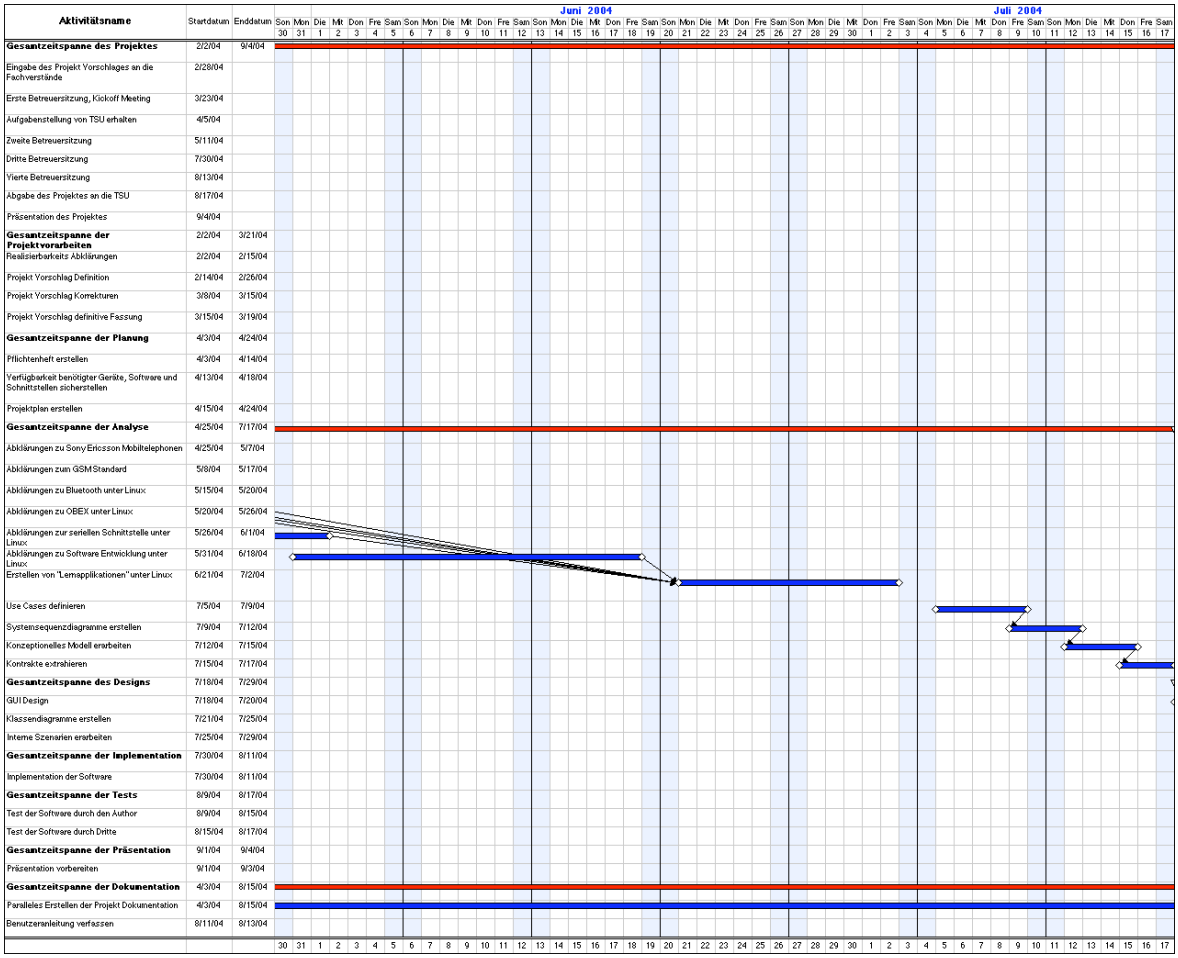
3.4.1 Projektplan Februar bis März



3.4.2 Projektplan April bis Mai

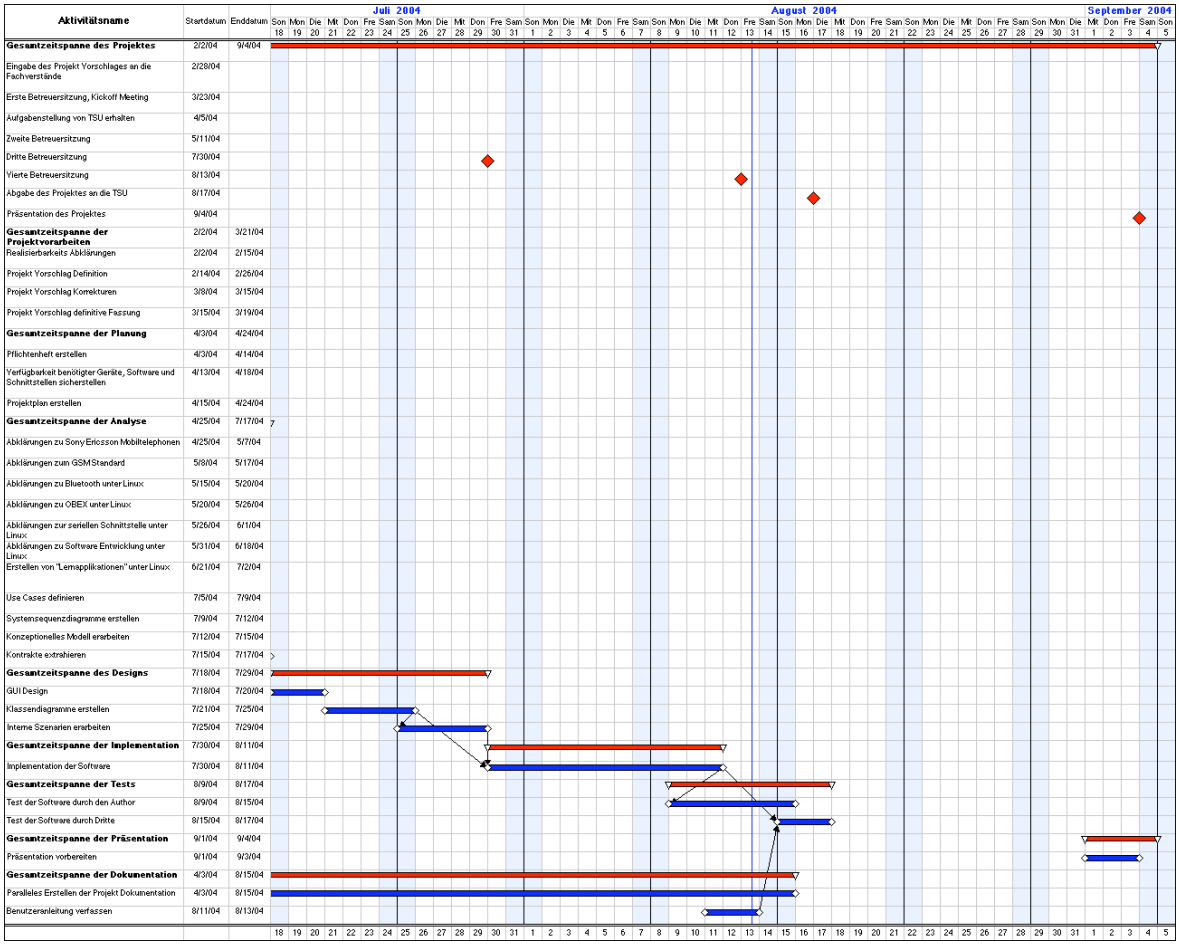


3.4.3 Projektplan Juni bis Juli



Aktivitätsname	Startdatum	Enddatum
Gesamtzeitspanne des Projektes	2/2/04	9/4/04
Eingabe des Projekt Vorschlages an die Fachverstände	2/28/04	
Erste Betreuersitzung, Kickoff Meeting	3/23/04	
Aufgabenstellung von TSU erhalten	4/5/04	
Zweite Betreuersitzung	5/11/04	
Dritte Betreuersitzung	7/30/04	
Vierte Betreuersitzung	8/13/04	
Abgabe des Projektes an die TSU	8/17/04	
Präsentation des Projektes	9/4/04	
Gesamtzeitspanne der Projektvorarbeiten	2/2/04	3/21/04
Realisierbarkeits Abklärungen	2/2/04	2/15/04
Projekt Vorschlag Definition	2/14/04	2/26/04
Projekt Vorschlag Korrekturen	3/8/04	3/15/04
Projekt Vorschlag definitive Fassung	3/15/04	3/19/04
Gesamtzeitspanne der Planung	4/3/04	4/24/04
Pflichtenheft erstellen	4/3/04	4/14/04
Verfügbarkeit benötigter Geräte, Software und Schnittstellen sicherstellen	4/13/04	4/18/04
Projektplan erstellen	4/15/04	4/24/04
Gesamtzeitspanne der Analyse	4/25/04	7/17/04
Abklärungen zu Sony Ericsson Mobiltelephonen	4/25/04	5/7/04
Abklärungen zum GSM Standard	5/8/04	5/17/04
Abklärungen zu Bluetooth unter Linux	5/15/04	5/20/04
Abklärungen zu OBEX unter Linux	5/20/04	5/26/04
Abklärungen zur seriellen Schnittstelle unter Linux	5/26/04	6/1/04
Abklärungen zu Software Entwicklung unter Linux	5/31/04	6/18/04
Erstellen von "Lernapplikationen" unter Linux	6/21/04	7/2/04
Use Cases definieren	7/5/04	7/9/04
Systemsequenzdiagramme erstellen	7/9/04	7/12/04
Konzeptionelles Modell erarbeiten	7/12/04	7/15/04
Kontrakte extrahieren	7/15/04	7/17/04
Gesamtzeitspanne des Designs	7/18/04	7/29/04
GUI Design	7/18/04	7/20/04
Klassendiagramme erstellen	7/21/04	7/25/04
Interne Szenarien erarbeiten	7/25/04	7/29/04
Gesamtzeitspanne der Implementation	7/30/04	8/11/04
Implementation der Software	7/30/04	8/11/04
Gesamtzeitspanne der Tests	8/9/04	8/17/04
Test der Software durch den Author	8/9/04	8/15/04
Test der Software durch Dritte	8/15/04	8/17/04
Gesamtzeitspanne der Präsentation	9/1/04	9/4/04
Präsentation vorbereiten	9/1/04	9/3/04
Gesamtzeitspanne der Dokumentation	4/3/04	8/15/04
Paralleles Erstellen der Projekt Dokumentation	4/3/04	8/15/04
Benutzeranleitung verfassen	8/11/04	8/13/04

3.4.4 Projektplan August bis September



3.4.5 Vergleich „Soll – Ist“ der Projektplanung

Der Projektplan wurde grundsätzlich eingehalten und in die Realität umgesetzt.

Probleme enstanden im Juli.

Ich betreue für meinen Arbeitgeber Server mit diversen Diensten wie Mail, File, Web, Groupware, etc. für 500 Angestelle und 2000 Studenten. Diese Server und deren Dienste werden per Mitte September auf neue Hard- und Software migriert, was primär meine Aufgabe ist.
Da dieses Projekt sehr wichtig ist, bedarf es gründlicher Vorbereitung und der Koordination mit der zentralen Informatik meines Arbeitgebers.

Leider laufen meine beiden bisher grössten Projekte, diese Diplomarbeit zum Informatik Techniker und obenstehend beschriebene Migration, zeitlich fast exakt parallel.
So konnte ich im Juli geplante Ferien, in denen ich die Diplomarbeit vorantreiben wollte, nur eingeschränkt beziehen, was mich bezüglich des Diplomarbeit Projektes in arge terminliche Schwierigkeiten brachte. Es entstanden Abweichungen zum Projektplan, viele Aufgaben, die auf Mitte Juli geplant waren, konnten erst Anfangs August umgesetzt werden.

3.5 Tatsächlicher getätigter Aufwand

Diese Tabelle enthält die tatsächlich getätigten Aufwendungen pro Aufgabe im Vergleich zur Schätzung, die bei Projekteingabe gemacht wurde.

Aufgabe	Geschätzter Aufwand in Stunden	Getätigter Aufwand in Stunden
Vorabklärungen	15	30
Betreuersitzungen	10	5
Pflichtenheft erstellen	5	5
Analyse	30	50
Design	25	20
Implementation	30	30
Dokumentation	50	50
Tests	15	15
Korrekturen Software	10	5
Korrekturen Dokumentation	5	5
Bedienungsanleitung	5	5
Unvorhergesehenes	10	0
Präsentation vorbereiten	5	5
Benötigte Gesamtzeit	**215**	**225**

Nachfolgend einige Erläuterungen zu den entstandenen Differenzen.

Die Vorabklärungen haben doppelt so viel Zeit beansprucht, als vorgesehen. Der Hauptgrund liegt darin, dass der Author die Zusammenhänge und Komplexität einer Mobiltelephon – PC Kommunikation unterschätzt hat. Es mussten 15h mehr als geplant investiert werden, um einen Überblick über die Materie zu erhalten und aus diesen Erkenntnissen ein Pflichtenheft zu erstellen.

Die Sitzungen mit meinem Betreuer waren kürzer als geplant, aber keinesfalls unwichtig oder gar durchgeboxt.

Die Analyse dauerte 20h mehr als projektiert, ebenfalls eine knappe Verdoppelung der benötigten Zeit. Die Gründe liegen auch hier einerseits in der Unterschätzung des Problems, andererseits aber auch die nichtexistente Erfahrung der Authors bezüglich Software Entwicklung unter Linux. Es musste viel Zeit aufgewendet werden, um dieses fehlende Wissen aufzubauen.

Das Design konnte etwas speditiver als geplant erledigt werden. Die Abweichung ist aber zu klein, als dass daraus Schlüsse gezogen werden könnten.

Es wurde nach der Implementation weniger Zeit als veranschlagt für Korrekturen an der Software benötigt. Eine mögliche Ursache ist, dass der Author während der Implementation konsequent getestet hat. Andererseits ist auch diese Abweichung gering.

Der Punkt „Unvorhergesehenes“ ist vollständig in der Planung und der Analyse aufgegangen.

Die tatsächlich benötigte Zeit ist um 10h höher als ursprünglich geplant.
Die Gründe sind dieselben, die obenstehend bei den Erläuterungen der Analyse aufgeführt worden sind.

4 Analyse

4.1 Vorabklärungen

4.1.1 Dokumentation zu Sony Ericsson Mobil Telephonen

Aktuelle Entwickler Dokumentationen zu Sony Ericsson Mobil Telephonen sind direkt via Sony Ericsson leider nicht zu beschaffen. Um Einsicht in diese Dokumente zu erhalten, müssten erst diverse Sachverhalte mit Sony Ericsson schriftlich geregelt werden. Ein Punkt tangiert die Bestimmungen der TSU betreffend Rechte an der Dokumentation. Es dürfte in der Dokumentation von BluePhone nichts enthalten sein, was auch in der Entwickler Dokumentationen von Sony Ericsson festgehalten ist.

Glücklicherweise ist die Dokumentation zu älteren Produkten von Sony Ericsson auf dem Netz verfügbar, mit Erlaubniss der Herstellerfirma. So fand ich unter http://renegade.w3xs.com/dev.htm den AT Befehlssatz zum T68i, dem Vorgängermodell meines T610, das ich zur Entwicklung von BluePhone verwendet habe.

Das Dokument heisst „Ericsson T68i AT Command Set". Es umfasst 222 Seiten und würde den Rahmen dieser Dokumentation trotz seiner zentralen Stellung in der Entwicklung von BluePhone sprengen und wird deshalb nicht in dieses Dokument eingebunden.

Da das Betriebsystem des T610 eine Weiterentwicklung dessen des T68i ist, konnte davon ausgegangen werden, dass die im „Ericsson T68i AT Command Set" dokumentierten Funktionen auch für die neueren Modelle der T-Serie Gültigkeit haben. Diese Annahme erwies sich als richtig.

4.1.2 Dokumentation zum GSM Standard

Der GSM Standard definiert grundsätzlich die Schnittstellen von Mobilen Kommunikationsgeräten, somit auch von Mobiltelephonen.

Von besonderem Interesse für dieses Projekt ist der ETSI-GSM-AT-Befehlssatz. Dieser beschreibt alle AT Befehle, welche ein Mobiltelephon beherrschen sollte. Sollte, weil dies leider nicht für alle Befehle auf alle Hersteller zutrifft.
Ich verwende für dieses Projekt den GSM Standard 07.07 in der Version 5.0.0.

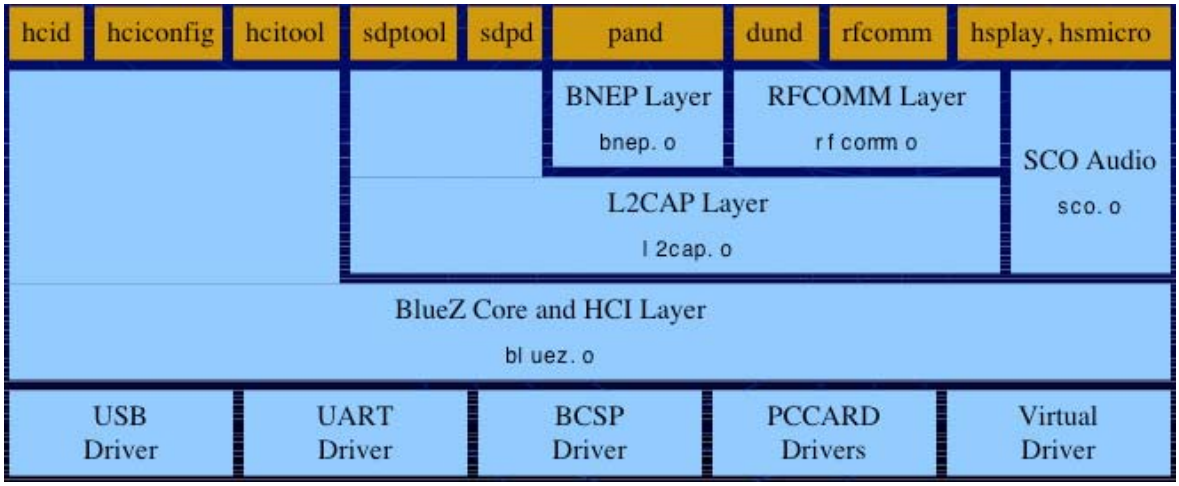
Der erste GSM Standard in heutigem Sinne wurde ungefähr 1994 von der 1987 gegründeten „GSM Association" veröffentlicht. Die „GSM Association" ist eine Handelsvereinigung bestehend aus über 600 Herstellern der Kommunikationsindustrie.

In Europa wird der Standard vom ETSI herausgegeben.

4.1.3 Dokumentation zu Bluetooth unter Linux

Bluetooth 1.1 wurde Anfangs 2001 von IrDA (Infrared Data Association) spezifiziert, seit Mitte 2002 ist das Bluetooth Layer fester Bestandteil des Kernels. Implementiert hat diese Kernel Module das Projekt BlueZ. Der aktuelle Kernel 2.4.25 unterstützt sehr viele Protokolle, die auf Bluetooth aufbauen. Für BluePhone vor allem von Interesse sind HCI, SDP, RFCOMM und OBEX. Auf diese Protokolle wird später genauer eingegangen werden.

Die folgende Abbildung beschreibt den Aufbau des Bluetooth Subsystems unter Linux.



Linux Bluetooth Subsystem
Hellblaue Bereiche befinden sich im Kernelspace, braune im Userspace.

BluePhone benötigt für eine Bluetooth Verbindung zum Mobiltelephon rfcomm. Dies stellt einen Socketstream für den Datenaustausch sowie ein TTY Device (Serielles Gerät) zur Verfügung. Das TTY Device wird von Bluephone intensiv genutzt, um dem Mobiltelephon AT Befehle zu übermitteln und dessen Antworten zu empfangen. RFCOMM stellt dieselben Dienste auch für ein IrDA Gerät zur Verfügung.

Der hcid (Host Controller Interface Daemon) handhabt die Kommunikation mit dem Bluetooth Device des Rechners, um Userspace Programmen Zugriff auf tiefere Bluetooth Layer zu ermöglichen. Er bietet eine einheitliche Schnittstelle zur Bluetooth Firmware und gehört eigentlich nicht direkt zum Bluetooth Protokoll Stack, aber es wird unter anderem dazu verwendet L2CAP Pakete an den Link Manager der Firmware zu senden und ist somit der unterste Layer, der im Kernel implementiert ist. HCI dient außerdem dazu, die aktuelle Configuration und Features sowie den Status des Geräts auszulesen und zu setzen. Die Kommunikation ist paket- und verbindungsorientiert. Dieser Dienst wird bei einigermassen aktuellen Distributionen automatisch gestart und erkennt zur Laufzeit (Hotplug) neue Geäte wie z.B. einen USB Bluetooth Dongle.

Die hcitools werden benötigt, um Informationen von Bluetooth Geräten, lokalen wie entfernten, zu gewinnen. Diese Informationen sind unter anderem die BTADD (Bluetooth Address, 6 Byte gross, ähnlich MAC Adresse einer NIC) oder der Name eines Gerätes.

Ein Anwendungsbeispiel der hcitools:

```
# hcitool scan
Scanning ...
        00:02:C7:1E:1D:B8       iPAQ H3970
        00:04:0E:21:06:FD       AVM BlueFRITZ! AP-X
        00:0A:D9:5C:75:57       Sony Ericsson T68i
        00:A0:57:AD:22:0F       ELSA Vianect Blue ISDN
        00:90:02:63:E0:83       Bluetooth Printer
        00:80:37:06:78:92       Ericsson T39m
        08:00:46:0E:CE:50       SONY Cyber-shot
        00:04:61:50:D4:3E       EPox BT-PM01B 80D43E
# hcitool info 00:04:0E:81:06:FD
Requesting information ...
        BD Address:  00:04:0E:21:06:FD
        Device Name: AVM BlueFRITZ! AP-X
        LMP Version: 1.1 (0x1) LMP Subversion: 0x1
        Manufacturer: AVM Berlin (31)
        Features: 0x2f 0xbe 0x05 0x00
                <3-slot packets> <5-slot packets> <encryption> <slot offset>
                <role switch> <RSSI> <channel quality> <SCO link>
                <HV2 packets> <HV3 packets> <A-law log> <CVSD>
                <power control>
```

Der sdpd (Service Discovery Protocoll Daemon) dient der Abfrage der auf dem entfernten Gerät verfügbaren Dienste. Seine Dienste werden wiederum von OBEX und RFCOMM verwendet. Auch dieser Dienst startet bei aktuellen Distributionen automatisch.

Die SDP Dienste basieren auf dem L2CAP (Logical Link Control and Adaptation Protocol) Layer. Die Hauptaufgabe dieses Protokolls ist die Fragmentierung, das Groupmanagement und die Implementierung höherer Protokolle wie RFCOMM oder SDP.

Die sdptools stellen eine Userspace Schnittstelle zu den SDP Diensten bereit. Diese werden einerseits benötigt, um die Dienste eines lokalen SDP Daemons zu konfigurieren, andererseits um die Dienste eines entfernten Bluetooth Gerätes abzufragen.

Alle Linux Bluetooth Komponenten verfügen über Man Pages, die detailiertere Informationen zur Verfügung stellen.

4.1.4 Dokumentation der seriellen Schnittstelle unter Linux

Da BluePhone über ein serielles Gerät (rfcomm Device) AT Befehle absetzen muss, wird die serielle Schnittstelle des Linux Kernels verwendet.

Der Author hat wenig Erahrung im Umgang mit seriellen Schnittstellen unter C/C++ und gar keine unter Linux. Bei bei der Suche nach Dokumentation zu diesem Thema bin ich über ein interessantes Projekt names ezV24 gestossen. Diese Bibliothek bietet eine C++ API um unter Linux (indirekt auch unter Windows) einfach Zugriff auf serielle Geräte zu erhalten.

BluePhone verwendet ezV24 zur Kommunikation mit seriellen Geräten.

Ein nicht ganz reelles Anwendungs Beispiel zu ezV24:

```
/** die ezV24 Bibliothek wird verwendet */
#include <ezV24/ezV24.h>

/** serielles Device definieren */
QString ttyDevice = „/dev/rfcomm0“

/**
  * ezv24 Port oeffnen.
  * Es wird ein Zeiger auf ein spezielles Handle zurückgeliefert.
  * Dieses Handle wird bei allen kuenftigen Zugriffen auf ezV24
  * Verwendung finden.
  */
v24_port_t* = v24OpenPort(ttyDevice, V24_RTS_CTS);
```

ezV24 ist unter http://ezv24.sourceforge.net/api-html/TheReference.html ausführlich dokumentiert.

4.1.5 Dokumentation zu OpenOBEX

OpenOBEX ist die freie Implementierung des OBEX Protokolls.
OBEX steht für Object Exchange und ist wie Bluetooth von IrDA spezifiziert worden. Es wird von vielen mobilen Geräten wie PDA's und Telephonen unterstützt.

OpenOBEX ist eine C API zu OBEX.

Die Dokumentation der Bibliothek findet sich leider vor allem in den Headerfiles und in Form von Beispielprogrammen, die im Quelltext mit der Bibliothek mitgeliefert werden.

OpenOBEX unterstützt neben Bluetooth auch IrDA und teilweise auch serielle Kabelverbindungen als Transport Layer.

Ein sehr kurzes Code Beispiel zu OpenOBEX:

```
/** die C Bibliothek OpenOBEX wird verwendet */
extern "C"
{
  #include <openobex/obex.h>
}

/**
  * Enthaelt alle Daten einer Obex Session.
  */
struct BlueObex
{
      int connected;
      bool client_done;
      enum session_transport transport;
      enum session_operation operation;
      enum session_target target;
      const char *filename;
};

/**
  * Initialisiert ein Obex Objekt mit allen benoetigen Parametern.
  */
obex_t* obexInit(struct BlueObex *myObex,
                        int method, const char *address, int channel);
```

4.1.6 Kommunikation mit dem Mobiltelephon

Mit einem Mobiltelephon wird grundsätzlich, wie mit einem analogen Modem, über eine serielle Schnittstelle kommuniziert. Es werden AT Kommandos abgesetzt, auf die das Mobiltelephon antwortet. Diese Antworten können dann entgegengenommen und verarbeitet werden.

Die serielle Schnittstelle wird, wie unter Punkt 4.1.3 beschrieben, von rfcomm bereitgestellt.

Ein Beispiel zur Herstellung einer rfcomm Verbindung:

```
/**
  * Diese Methode verbindet das rfcomm Device mit dem
  * entfernten Bluetooth Device (dem Telephon).
  * Parameter: const QString& comDevice, btAddr
  */
int BlueLink::connectRfcomm(const QString& comDevice,
                            const QString& btAddr)
{
      /* setze das rfcomm Kommando zusammen. */
      m_rfcommCommand = "/usr/bin/rfcomm connect ";
      m_rfcommCommand.append(comDevice);
      m_rfcommCommand.append(" ");
      m_rfcommCommand.append(btAddr);
      m_rfcommCommand.append(" 1");

      /* execve will Char Arrays sehen, keine QStrings. */
      const char *commandChar = m_rfcommCommand.latin1();

      pid_t child_pid;
      int child_status;

      child_pid = fork();

      if(child_pid == 0)
      {
            /* Dieser Teil wird vom Kind Prozess ausgefuehrt. */

            execl("/bin/sh", "sh", "-c", commandChar, 0);

            /* Falls execve zurueckkommt, ist etwas schiefgegangen. */
            printf("Unknown command\n");
            exit(0);
      }
      else
      {
            /* Dieser Teil wird vom Vater Prozess ausgefuehrt. */

            return child_status;
      }
}
```

Ein Problem beim Datenaustausch ist der von Mobiltelephonen normalerweise verwendete 7 Bit Zeichensatz GSM. Um Daten vom Mobiltelephon zu empfangen, muss dieser Zeichensatz jeweils in einen Zeichensatz konvertiert werden, den ein PC verarbeiten kann. Umgekehrt dasselbe, sollen Daten an das Mobiltelephon geschickt werden, müssen diese in das GSM Format übersetzt werden.

Dies ist der obenstehend erwähnte 7 Bit Zeichensatz GSM:

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	1	0	1	0	1
b4	b3	b2	b1	Hex	0	1	2	3	4	5	6	7
0	0	0	0	0	@	Ä	SP	0	¡	P	¿	p
0	0	0	1	1	£	_	!	1	A	Q	a	q
0	0	1	0	2	$	Ö	"	2	B	R	b	r
0	0	1	1	3	¥	Ã	#	3	C	S	c	s
0	1	0	0	4	è	Ë	¤	4	D	T	d	t
0	1	0	1	5	é	Ù	%	5	E	U	e	u
0	1	1	0	6	ù	Ð	&	6	F	V	f	v
0	1	1	1	7	ì	Ø	'	7	G	W	g	w
1	0	0	0	8	ò	Ó	(	8	H	X	h	x
1	0	0	1	9	ç	È	)	9	I	Y	i	y
1	0	1	0	A	LF	Î	*	:	J	Z	j	z
1	0	1	1	B	Ø	1)	+	;	K	Ä	k	ä
1	1	0	0	C	ø	Æ	,	<	L	Ö	l	ö
1	1	0	1	D	CR	æ	-	=	M	Ñ	m	ñ
1	1	1	0	E	Å	ß	.	>	N	Ü	n	ü
1	1	1	1	F	å	É	/	?	O	§	o	à

Ein weiterer Knackpunkt sind Kurzmitteilungen.

Leider unterstützen die meisten Mobiltelephone nicht die komfortable Art der Übergabe von Textnachrichten im Textmodus, also als Klartext.
Die Entwickler des GSM Standards haben dazu den PDU-Modus geschaffen.

Die an das Telefon zu übergebende PDU zum Versand einer Kurznachricht besteht im wesentlichen aus zwei Teilen:

- der Adresse des zu verwendenden SMSC (Short Message Service Center, optional)
- der eigentlichen SMS-SUBMIT-PDU, wie sie auch im Netz übertragen wird

Eine PDU setzt sich wie folgt zusammen:

`07 91947122723033 25 00 0C 91947112325476 00 00 0B D4F29C4E2FE3E9BA4D19`

oder:

`00 25 00 0C 91947112325476 00 00 0B D4F29C4E2FE3E9BA4D19`

Erläuterung der einzelnen PDU-Elemente:

SMSC Adresse:

`07 91947122723033`

Im ersten Byte dieses Informationselements steht die Länge der gelieferten Adressdaten, also die Länge der SMSC-Adresse, in Bytes, in oben genanntem Beispiel also 7 Bytes.
Im zweiten Byte steht der TOA (Type of address) der gelieferten SMSC-Adresse.
Es existieren zwei relevante Werte für einen TOA: 0x91 für Nummern im internationalen Format oder 0x81 für Nummern im nationalen Format.
Die restlichen Bytes enthalten BCD-kodiert die Nummer des SMSC das verwendet werden soll. Die Nummer beginnt im zweiten Halbbyte des ersten Bytes, ein eventuell nicht benutztes Halbbyte an der letzten Stelle muss 0xF enthalten.

Im zweiten Beispiel enthält die SMSC Adresse nur:

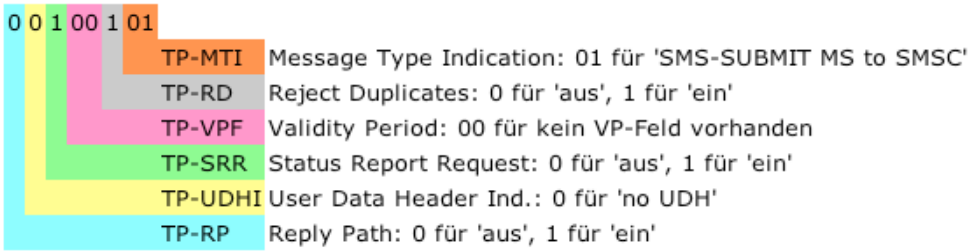
`00`

Erklärung, wenn im Telefon bereits eine SMSC-Nummer konfiguriert ist, kann die Angabe der SMSC-Adresse entfallen. Hierzu wird einfach als Längenbyte 0x00 übergeben.

Message Flags:

`25`

Effizienterweise sind hier in einem Byte gleich 6 Parameter untergebracht:



Message Reference Number:
00
Eine lokal verwendete Referenznummer für die zu verschickende Kurznachricht. Zustellbestätigungen oder Fehlermeldungen enthalten diese Referenznummer und können damit der ursprünglichen Nachricht zugeordnet werden. Das Telefon vergibt selbständig eine eigene Referenznummer, falls keine angegeben wird, deshalb hier 0x00.

Destination Address:
0C 91947112325476
Das erste Byte dieses Informationselements gibt die Anzahl Ziffern der Zieladresse an. Der Rest der Nummer wird wie die SMSC Addresse kodiert.

Protocol Identifier:
00
Gibt an, welches Protokoll den verschickten Daten zugrunde liegt. Ein simples 0x00 genügt im Allgemeinen hier, was den Defaultwert darstellt.

Data Coding Scheme:
00
Dieses Byte teilt dem empfangenden Mobiltelephon mit, wie die Daten der Kurznachricht zu decodieren sind. Ein simples 0x00 genügt auch hier.

User Data:
0B D4F29C4E2FE3E9BA4D19
Wieder ein Längenbyte, das erste Byte gibt an, wie viele Zeichen Nutzdaten folgen, und zwar nach dem decodieren der Nachricht. Die folgenden Bytes enthalten dann die effektive Nachricht.

4.2 Softwareentwicklung unter Linux

4.2.1 Übersicht

Grundsätzlich gestaltet sich Softwareentwicklung unter Linux nicht unterschiedlich wie unter kommerziellen Betriebsystemen wie Windows oder Mac OS X.

Ein grosser Unterschied ist, es gibt eine grosse Qual der Wahl. Von der verwendeten Graphikbibliothek bis zur benutzten Entwicklungsumgebung, überall ist der Entwickler frei bezüglich dem, das er verwenden möchte. Für einen Einsteiger wie mich eher ein Hindernis als eine Bereicherung.
Nachfolgend möchte ich kurz auf die Komponenten eingehen, die ich verwendet habe.

4.2.2 QT

QT ist ein, für Open Source Projekte frei verfügbares, komplettes C++ Applikations Framework mit einer Klassenbibliothek der Firma Trolltech.

Für mich ist QT aus folgenden Gründen besonders interessant:

- sehr gute Online Dokumentation aller Klassen
- Windows ähnliche, mir aus dem Unterricht der TSU vertraute Graphik Widgets
- Plattform übergreifend, u.a. unterstützt QT Windows, Linux oder Mac OS X
- Neben der Borland Suite einziges Produkt, das unter Linux eine graphische Entwicklung des GUIs per Drag&Drop ermöglicht.

Weitere Informationen zu QT sind Online verfügbar unter:
http://www.trolltech.com/products/qt/index.html
Die Dokumentation zu QT findet sich Online unter:
http://doc.trolltech.com/3.3/index.html

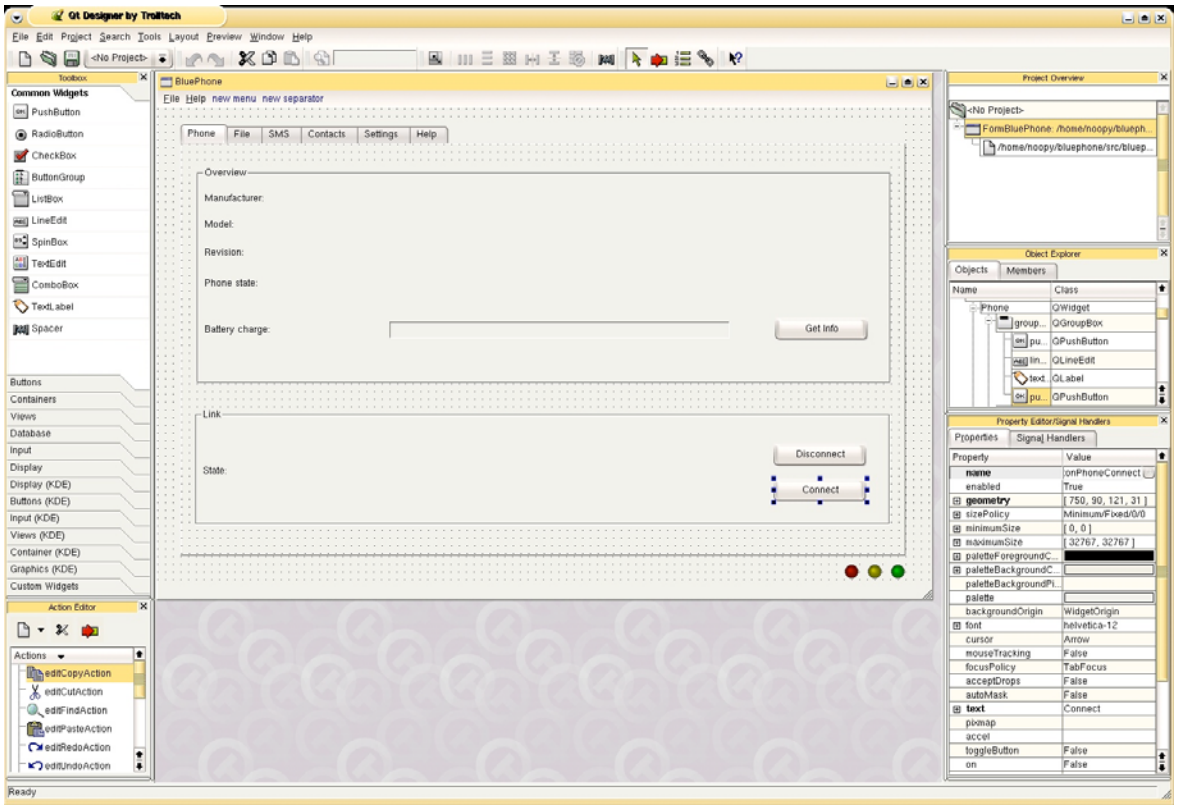
4.2.3 QT Designer

Als ich mich nach Wegen umschaute, unter Linux ein benutzerfreundliches GUI zu gestalten, las ich immer wieder: vi (Konsolen Text Editor)
Ich dachte mir, es kann doch nicht sein, dass im Jahre 2004 alle Entwickler, die unter Linux arbeiten, ihre GUIs direkt in einem Texteditor kodieren. Es ist zum Glück nicht so.

Die Firma Trolltech stellt für Ihre QT Bibliothek einen Werkzeug names „QT Designer“ zur Vefügung, das eine graphische Entwicklung eines GUIs ermöglicht. QT Designer kann von einer Person, die schon eimal ein GUI in Microsofts „Visual Studio“ oder Borlands „Builder“ entworfen hat, intuitiv bedint werden, weshalb ich nicht weiter auf den QT Designer eingehen werden.

Weitere Informationen zum QT Designer sind Online verfügbar unter:
http://www.trolltech.com/products/qt/designer.html

Ein Screenshot des QT Designers:



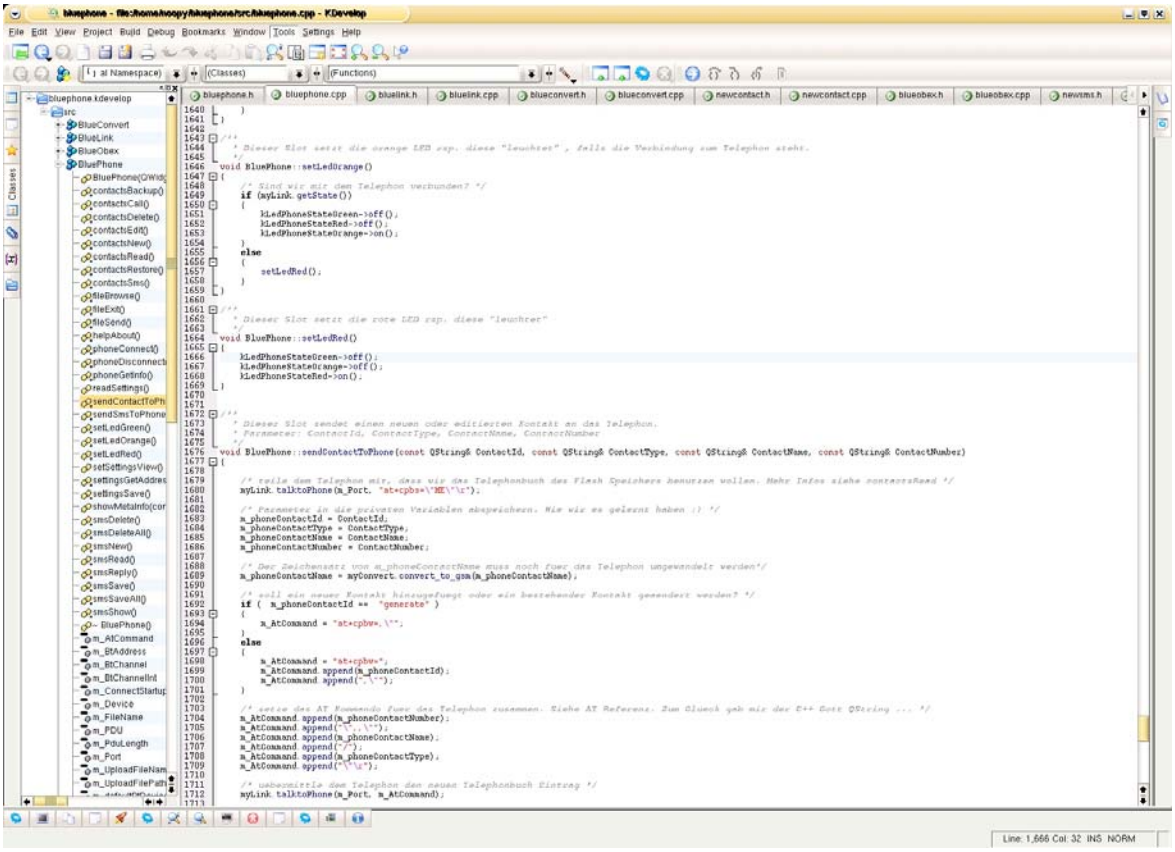
4.2.4 Entwicklungsumgebung KDevelop

Auch auf der Suche nach einer geeigneten IDE las ich des öftern: vi
Doch auch auf diesem Gebiet hat die Gegenwart Linux eingeholt, trotz des passiven Widerstandes einer „alten Schule“ von Entwicklern, die sich mit freier Software befassen.

KDevelop ist eine freie, dem „Visual Studio“ von Microsoft sehr ähnliche, komplette und einfach zu bedienende Entwicklungsumgebung. Für mich ist die Nähe zur Bedienung von Visual Studio ein grosser Vorteil, da an der TSU der C++ Unterricht ausschlisslich unter Visual Studio erteilt wurde. Auch auf Kdevelop werde ich deshalb nicht weiter eingehen.

Weitere Informationen zu KDevelop sind Online verfügbar unter: http://kdevelop.org/

Ein Screenshot von KDevelop:



4.2.5 Doxygen

Doxygen ist ein Dokumentationssystem für C++, C, Java, Objectiv-C und teilweise auch für PHP, C # und D.

Es erzeugt direkt aus den Quellcode Kommentaren eine Dokumentation einer Software respektive einer Klasse. Meistens wird es für API Dokumentationen verwendet, es ist aber auch als Werkzeug zur Dokumentation einer „normalen“ Software geeignet. Doxygen kann diverse Formate erzeugen, unter anderem HTML, LATEX (somit indirekt auch Postscript oder PDF Dateien), RTF und viele mehr. Für mich von Interesse ist HTML als Ausgabeformat.

Mit Doxygen kann ich eine stets aktuelle Dokumentation von BluePhone per Knopfdruck generieren. Eine Doxygen generierte Klassen Dokumentation von BluePhone ist auf der, dieser Dokumentation beigelegten, CD enthalten. Siehe Punkt 11.5.1 für genauere Informationen diesbezüglich.

Doxygen wird unter Linux entwickelt, läuft aber auch auf sehr vielen anderen Unix Varianten sowie unter Windows 9x/NT und Mac OS X.

Weitere Informationen zu Doxygen sind Online verfügbar unter:
http://www.stack.nl/~dimitri/doxygen/

4.3 Systemgrenzen

4.3.1 Allgemein

Die Systemgrenzen werden grundsätzlich gemäss Pflichtenheft gezogen.

4.3.2 Auflistung

BluePhone ist somit verantwortlich für folgende Operationen:

- Den Verbindungsaufbau zum Telephon unter Zuhilfnahme der unter Linux zu diesem Zweck verfügbaren Mittel
- Das Trennen der Verbindung unter Zuhilfnahme der unter Linux zu diesem Zweck verfügbaren Mittel
- Das Erstellen von Befehlen für das Mobiltelephon
- Das Versenden dieser Befehle an das Telephon
- Die Darstellung eines GUI
- Die Entgegennahme vom Befehlen via GUI
- Die Verarbeitung von Befehlen, die via GUI vom Benutzer gegeben werden
- Die Entgegennahme von Daten via GUI
- Die Verarbeitung von Daten, die via GUI vom Benutzer geliefert werden
- Die Entgegennahme von Daten vom Telephon
- Die Verarbeitung von Daten, die vom vom Telephon geliefert wurden
- Die Darstellung von Daten, die vom vom Telephon geliefert wurden, im GUI

4.4 Aktoren

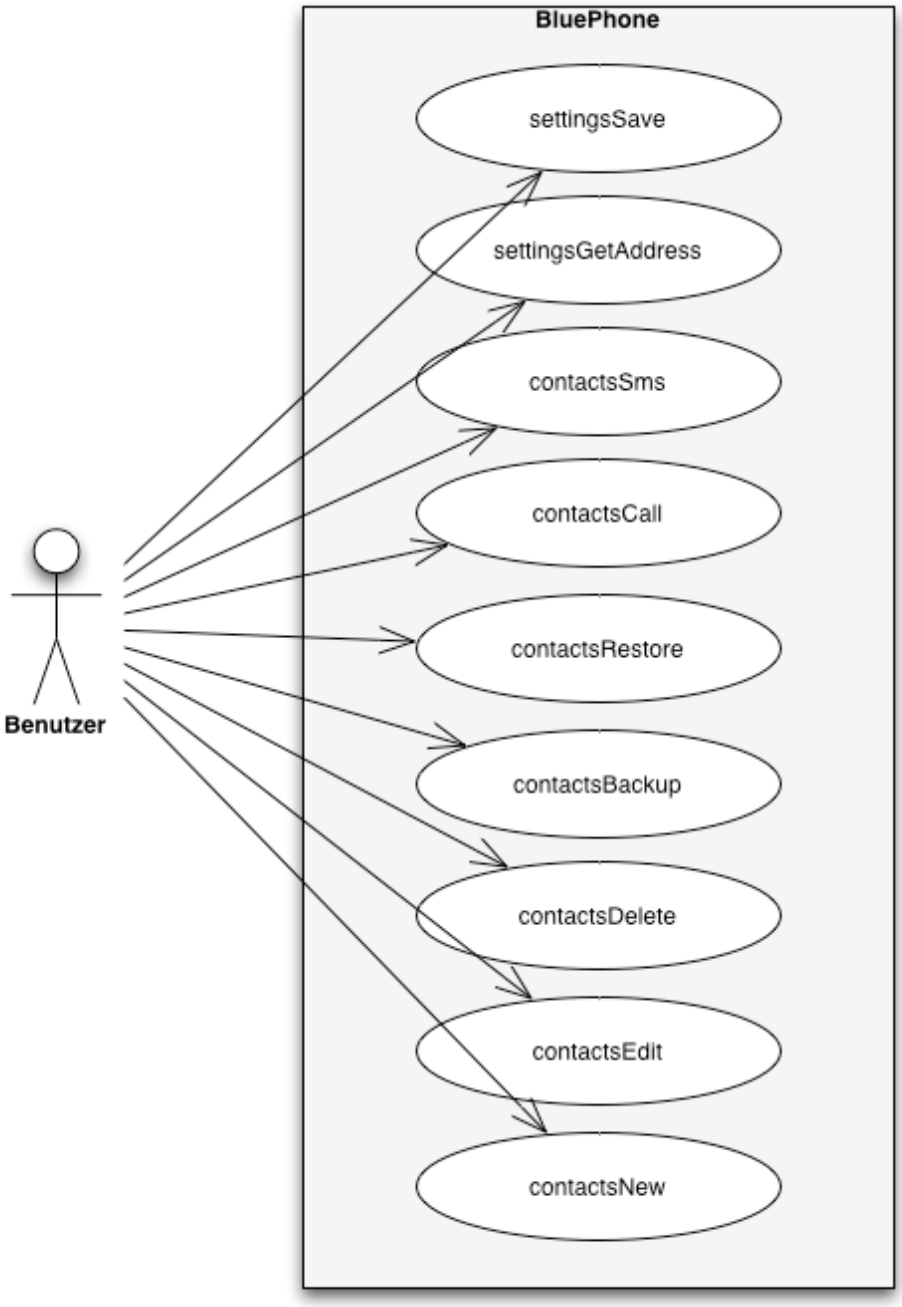
4.4.1 Beschreibung der Aktoren

Das System kennt zwei Aktoren.

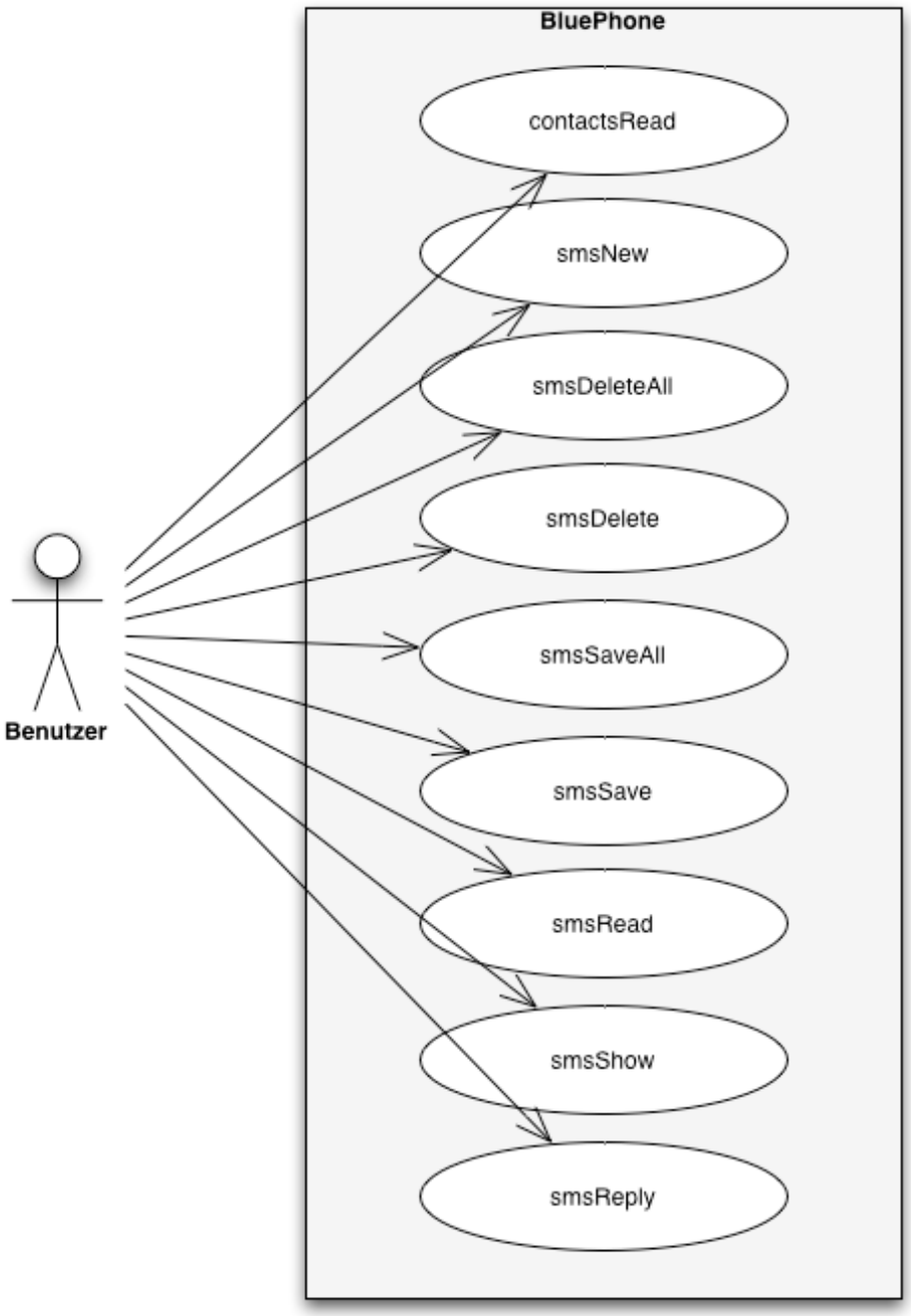
Benutzer	Steuert sämtliche Funktionen von BluePhone, erteilt alle Kommandos an Bluephone.
BlueSystem	Steuert die BluePhone internen Abläufe, die als Reaktionen auf Aktionen des GUIs respektive des Benutzers ausgelöst werden müssen.

4.5 Use Case Diagramme

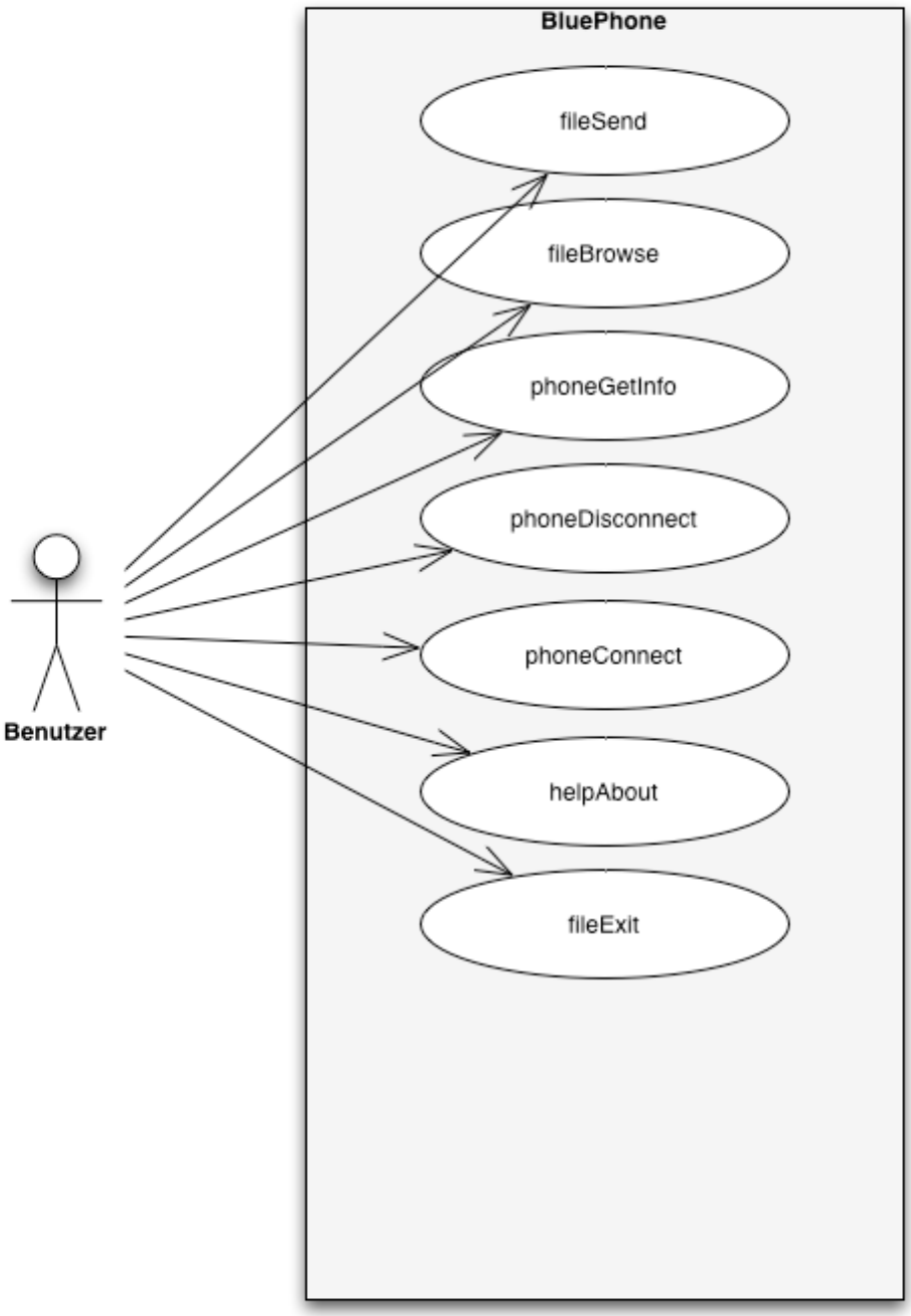
4.5.1 Use Case Diagramm erster Teil



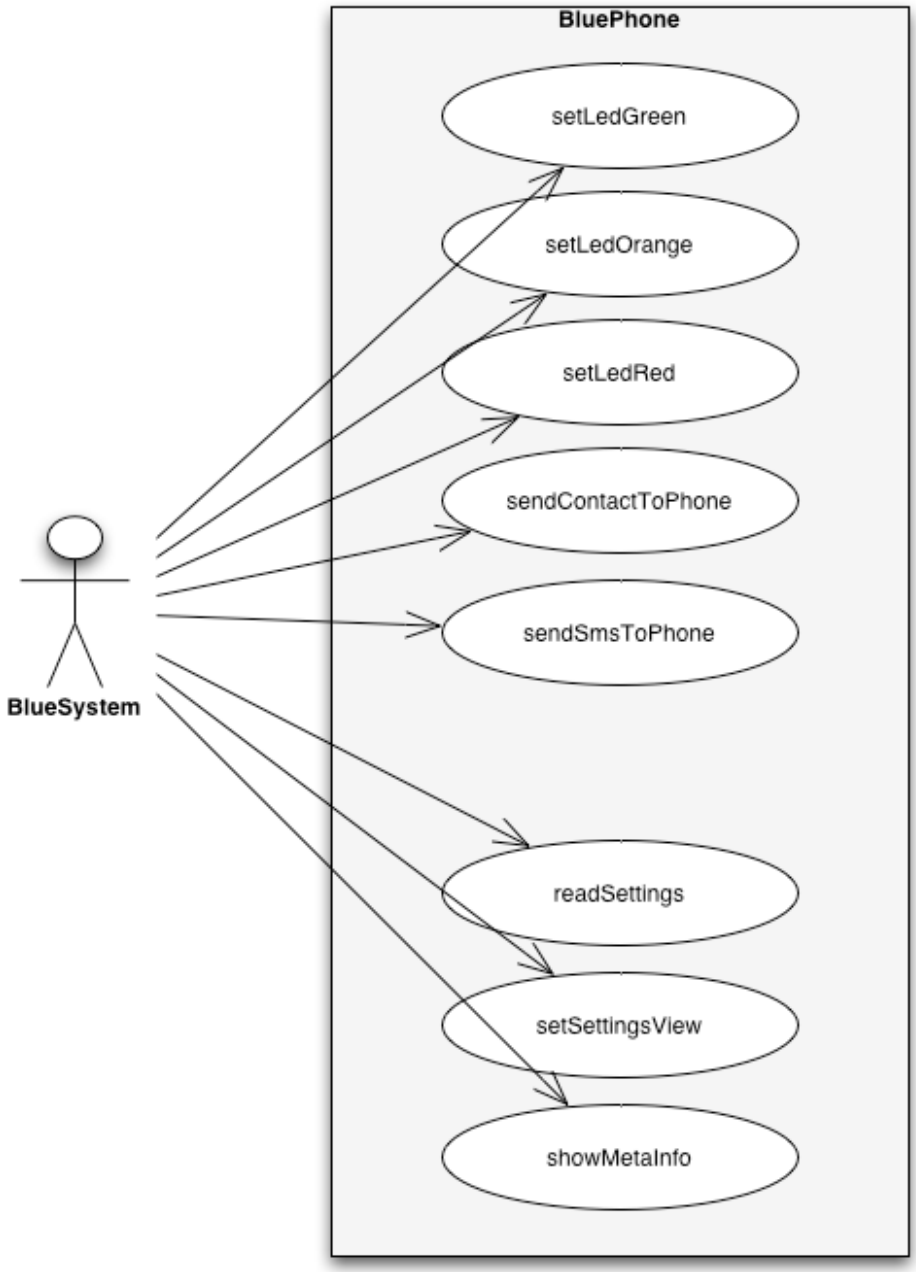
4.5.2 Use Case Diagramm zweiter Teil



4.5.3 Use Case Diagramm dritter Teil



4.5.4 Use Case Diagramm vierter Teil



4.6 Beschreibung der Use Cases

Name:	settingsSave	
Aktor:	Bediener	
Zweck:	Speichern der Einstellungen	
Kurzbeschreibung:	Die Einstellungen des Settings Tabs von BluePhone werden in eine KDE-Konfiguratiosdatei gespeichert	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSettingsSave	2. Speichert die Einstellungen in Konfigdatei ab 3. Zeigt Info Messagebox 4. Trennt Bluephone vom Mobiltelephon, falls verbunden

Name:	settingsGetAddress	
Aktor:	Bediener	
Zweck:	Info Messagebox anzeigen	
Kurzbeschreibung:	Zeigt eine Info Message Box, wie der Benutzer die Bluetooth Adresse seines Mobiltelephons auslesen kann.	
Priorität:	Sekundär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonGetAddress	2. Zeigt InfoMessagebox

Name:	contactsSms	
Aktor:	Bediener	
Zweck:	Senden einer SMS	
Kurzbeschreibung:	Öffnet einen „New SMS“ Dialog aus dem Contacts Tab mit der Nummer des selektierten Kontaktes als Empfängernummer	
Priorität:	Sekundär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonContactsSms	2. Setzt die orange LED 3. Zeigt einen „New SMS“ Dialog mit der Nummer des selektierten Kontaktes als Empfängernummer.
	4. Benutzer klickt Send oder Cancel	5. Falls Benutzer Send geklickt hat, ruft sendSmsToPhone auf, falls Cancel, Abbruch 6. Setzt die grüne LED

Name:	contactsCall	
Aktor:	Bediener	
Zweck:	Anrufen eines Kontaktes	
Kurzbeschreibung:	Ruft den selektierten Kontakt an	
Priorität:	Sekundär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonContactsCall	2. Ruft den selektierten Kontakt an

Name:	contactsRestore	
Aktor:	Bediener	
Zweck:	Wiederherstellen der Kontakte	
Kurzbeschreibung:	Stellt die Kontaktliste auf dem Mobiltelephon aus einem vorzeitig gesicherten Kontakte Backup wieder her.	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonContactsRestore	2. Setzt die orange LED 3. Datei Auswählen Dialog anzeigen
	4. Wählt Backup Datei aus	5. Abfragedialog, ob der Benutzer wirklich all seine Kontakte auf dem Mobiltelephon mit dem selektierten Backup überschreiben will.
	6. Klickt Yes oder No	7. Falls Benutzer Yes geklickt hat, Überschreiben aller Kontake auf dem Mobiltelephon mit dem Inhalt der selektierten Backupdatei und liest danach die Kontaktliste des Mobiltelephons ein Falls Benutzer No geklickt hat, Abbruch 8. Setzt die grüne LED

Name:	contactsBackup	
Aktor:	Bediener	
Zweck:	Sichern der Kontakte	
Kurzbeschreibung:	Sichert die Kontaktliste des Moblitelephons in eine Backup Datei	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonContactsBackup 3. Wählt Backup Datei aus	2. Zeigt Datei Auswählen Dialog an 4. Sichert die Kontaktliste des Mobiltelephons in die ausgewählte Backup Datei

Name:	contactsDelete	
Aktor:	Bediener	
Zweck:	Löschen eines Kontaktes	
Kurzbeschreibung:	Der selektierte Kontakt soll auf dem Mobiltelephon gelöscht werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonContactsDelete 4. Klickt Yes oder No	2. Setzt die gelbe LED 3. Abfragedialog, ob der Benutzer den selektierten Kontakt wirklich löschen will 5. Falls Benutzer Yes geklickt hat, löscht den selektierten Kontakt auf dem Mobiltelephon und liest danach die Kontaktliste des Mobiltelephons erneut ein Falls Benutzer No geklickt hat, Abbruch 6. Setzt die grüne LED

Name:	contactsEdit	
Aktor:	Bediener	
Zweck:	Kontakt editieren	
Kurzbeschreibung:	Der selektierte Kontakt soll auf dem Mobiltelephon editiert werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonContactsEdit oder doppelklickt auf einen Eintrag in listViewContacts	2. Setzt die orange LED 3. Zeigt Edit Contact Dialog mit den Werten des selektierten Kontaktes an
	4. Benutzer klickt Ok oder Cancel	5. Falls Benutzer Ok geklickt hat, Kontakt auf dem Mobiltelephon editieren Falls Benutzer Cancel geklickt hat, Abbruch 6. Setzt die grüne LED

Name:	contactsNew	
Aktor:	Bediener	
Zweck:	Neuen Kontakt erstellen	
Kurzbeschreibung:	Ein neuer Kontakt soll auf dem Mobiltelephon erstellt werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonContactsNew	2. Setzt die orange LED Zeigt New Contact Dialog an
	4. Benutzer klickt Ok oder Cancel	5. Falls Benutzer Ok geklickt hat, Kontakt auf dem Mobiltelephon speichern Falls Benutzer Cancel geklickt hat, Abbruch 6. Setzt die grüne LED

Name:	contactsRead	
Aktor:	Bediener	
Zweck:	Kontaktliste einlesen	
Kurzbeschreibung:	Die Kontaktliste des Mobiltelephons soll eingelesen und in listViewContacts dargestellt werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonContactsRead	2. Setzt die orange LED 3. Die Kontaktliste des Mobiltelephons wird eingelesen und in listViewContacts dargestellt 4. Setzt die grüne LED

Name:	smsNew	
Aktor:	Bediener	
Zweck:	Neues SMS verschicken	
Kurzbeschreibung:	Es soll ein neues SMS erstellt und veschickt werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSmsNew 4. Benutzer klickt Send oder Cancel	2. Setzt die orange LED 3. Zeigt einen „New SMS“ Dialog an 5. Falls Benutzer Send geklickt hat, ruft sendSmsToPhone auf Falls Benutzer Cancel geklickt hat, Abbruch 6. Setzt die grüne LED

Name:	smsDeleteAll	
Aktor:	Bediener	
Zweck:	Löschen aller SMS	
Kurzbeschreibung:	Sämtliche SMS auf dem Mobiltelephon sollen gelöscht werden.	
Priorität:	Sekundär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSmsDeleteAll	2. Setzt die orange LED 3. Zeigt Abfrage Dialog an, ob der Benutzer wirklich alle SMS auf dem Mobiltelephon löschen will
	4. Benutzer klickt Yes oder No	5. Falls Benutzer Yes geklickt hat, löscht alle SMS auf dem Mobiltelephon und liest die SMS Liste neu ein. Falls Benutzer No geklickt 6. hat, Abbruch Setzt die grüne LED

Name:	smsDelete	
Aktor:	Bediener	
Zweck:	Löschen einer SMS	
Kurzbeschreibung:	Ein SMS auf dem Mobiltelephon soll gelöscht werden.	
Priorität:	Sekundär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSmsDelete	2. Setzt die orange LED 3. Zeigt Abfrage Dialog an, ob der Benutzer wirklich das selektierte SMS auf dem Mobiltelephon löschen will
	4. Benutzer klickt Yes oder No	5. Falls Benutzer Yes geklickt hat, löscht das selektierte SMS auf dem Mobiltelephon und liest die SMS Liste neu ein. Falls Benutzer No geklickt hat, Abbruch 6. Setzt die grüne LED

Name:	smsSaveAll	
Aktor:	Bediener	
Zweck:	Sichern aller SMS	
Kurzbeschreibung:	Alle SMS auf dem Mobiltelephon sollen in eine Textdatei abgespeichert werden.	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSmsSaveAll 3. Wählt Datei zum Sichern aller SMS aus	2. Zeigt Datei Auswählen Dialog an 4. Sichert alle SMS in die ausgewählte Text Datei

Name:	smsSave	
Aktor:	Bediener	
Zweck:	Sichern eines SMS	
Kurzbeschreibung:	Ein SMS des Mobiltelephons soll in eine Textdatei abgespeichert werden.	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSmsSave 3. Wählt Datei zum Sichern des SMS aus	2. Zeigt Datei Auswählen Dialog an 4. Sichert das selektierte SMS in die ausgewählte Text Datei

Name:	smsRead	
Aktor:	Bediener	
Zweck:	Lesen der SMS Liste	
Kurzbeschreibung:	Alle auf dem Mobiltelephon vorhandenen SMS sollen gelesen und in listViewSms angezeigt werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSmsRead	2. Setzt die orange LED 3. Alle auf dem Mobiltelephon vorhandenen SMS werden gelesen und in listViewSms angezeigt 4. Setzt die grüne LED

Name:	smsShow	
Aktor:	Bediener	
Zweck:	Anzeigen einer SMS	
Kurzbeschreibung:	Soll ein selektiertes SMS in einer MessageBox anzeigen	
Priorität:	Sekundär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSmsView oder doppelklickt auf einen Eintrag in listViewSms	2. Zeigt das selektierte SMS zwecks besserer Lesbarkeit in einer MessageBox an

Name:	smsReply	
Aktor:	Bediener	
Zweck:	Beantworten einer SMS	
Kurzbeschreibung:	Es soll eine Antwort auf ein SMS erstellt und veschickt werden	
Priorität:	Sekundär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonSmsReply	2. Setzt die orange LED 3. Zeigt einen „Reply SMS“ Dialog an und füllt die Sendernummer sowie den Text des selektierten SMS in in diesen Dialog ab
	4. Benutzer klickt Send oder Cancel	5. Falls Benutzer Send geklickt hat, ruft sendSmsToPhone auf Falls Benutzer Cancel geklickt hat, Abbruch 6. Setzt die grüne LED

Name:	fileSend	
Aktor:	Bediener	
Zweck:	Hochladen einer Datei	
Kurzbeschreibung:	Es soll eine Bild- oder iMusic- Datei vom PC auf das Mobiltelephon hochgeladen werden.	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonFileSend	2. Setzt die orange LED 3. Sendet die ausgewählte Datei an das Mobiltelephon 4. Setzt die grüne LED

Name:	fileBrowse	
Aktor:	Bediener	
Zweck:	Auswählen einer Datei	
Kurzbeschreibung:	Es soll ein Datei auswählen Dialog angezeigt werden um eine Datei zu selektieren, die danach mit fileSend auf das Mobiltelephon hochgeladen werden kann. Weiter sollen nach Selektion der Datei Metainformationen dieser wie Grösse, Typ und bei Bilddateien auch die Abmessungen des Bildes in Pixeln angezeigt werden.	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonFileBrowse	
		2. Zeigt einen Datei Auswählen Dialog an
	3. Wählt auf das Mobiltelephon zu ladende Datei aus	
		4. Zeigt den Pfad der auf das Mobiltelephon zu ladenden Datei sowie deren zukünftigen Namen auf dem Mobiltelephon an Weiter werden, sofern lesbar, Metainformationen der Datei wie Grösse, Typ und bei Bilddateien auch die Abmessungen des Bildes in Pixeln angezeigt

Name:	phoneGetInfo	
Aktor:	Bediener	
Zweck:	Informationen des Mobiltelephons anzeigen	
Kurzbeschreibung:	Es sollen Statusinformationen des Mobiltelephons angezeigt werden. Diese sind im Detail: Hersteller, Modell und Revision des Mobiltelephons, der Status dessen sowie dessen Batterieladezustand	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonPhoneGetInfo	
		2. Setzt die gelbe LED 3. Zeigt Statusinformationen des Mobiltelephons an 4. Setzt die grüne LED

Name:	phoneDisconnect	
Aktor:	Bediener	
Zweck:	Trennen der Verbindung	
Kurzbeschreibung:	Die Verbindung zum Mobiltelephon soll getrennt werden. Weiter sollen alle Felder, die Daten des Mobiltelephons enthalten, geleert werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonPhoneDisconnect	2. Setzt die orange LED 3. Die Verbindung zum Mobiltelephon wird getrennt. Weiter werden alle Felder, die Daten des Mobiltelephons enthalten, geleert. 4. Setzt die rote LED

Name:	phoneConnect	
Aktor:	Bediener	
Zweck:	Herstellen der Verbindung	
Kurzbeschreibung:	Die Verbindung zum Mobiltelephon soll, unter Verwendung der unter Linux dazu verfügbaren Werkzeuge, hergestellt werden.	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf pushButtonPhoneConnect	2. Setzt die orange LED 3. Die Verbindung zum Mobiltelephon wird hergestellt 4. Setzt die grüne LED

Name:	helpAbout	
Aktor:	Bediener	
Zweck:	Anzeigen eines About Dialoges	
Kurzbeschreibung:	Es soll ein About Dialog angezeigt werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf helpAboutAction 3. Klickt auf Ok	2. Zeigt einen About Dialog an 4. Der About Dialog wird geschlossen

Name:	fileExit	
Aktor:	Bediener	
Zweck:	Programm beenden	
Kurzbeschreibung:	Das Programm soll beendet werden	
Priorität:	Primär	
Ablauf:	Bediener-Aktionen	Systemantwort
	1. Klickt auf fileExitAction	2. BluePhone wird beendet

Name:	setLedGreen	
Aktor:	BlueSystem	
Zweck:	Die grüne LED leuchtet	
Kurzbeschreibung:	Die grüne LED soll als Statusanzeige verdeutlichen, dass die Verbindung zum Mobiltelephon steht.	
Priorität:	Sekundär	
Ablauf:	BlueSystem-Aktionen	Systemantwort
	1. Überprüft die Verbindung zum Mobiltelephon	2. Falls die Verbindung steht, aktiviert die grüne LED, deaktiviert die orange und die rote LED

Name:	setLedOrange	
Aktor:	BlueSystem	
Zweck:	Die orange LED leuchtet	
Kurzbeschreibung:	Die orange LED soll als Statusanzeige verdeutlichen, dass Daten mit dem Mobiltelephon ausgetauscht werden. Da diese Datentransfers sehr lange dauern können, würde sich der Benutzer ohne ein solches Feedback der Applikation fragen, ob die Applikation abgestürzt ist oder ob noch ein Prozess im Gange ist.	
Priorität:	Sekundär	
Ablauf:	BlueSystem-Aktionen	Systemantwort
	1. Stellt Datentransfer zum Mobiltelephon fest	2. Falls die Verbindung steht, aktiviert die orange LED, deaktiviert die rote und die grüne LED

Name:	setLedRed	
Aktor:	BlueSystem	
Zweck:	Die rote LED leuchtet	
Kurzbeschreibung:	Die rote LED soll als Statusanzeige verdeutlichen, dass die Verbindung zum Mobiltelephon getrennt ist.	
Priorität:	Sekundär	
Ablauf:	BlueSystem-Aktionen	Systemantwort
	1. Überprüft die Verbindung zum Mobiltelephon	2. Falls die Verbindung 3. getrennt, aktiviert die rote 4. LED, deaktiviert die orange und die grüne LED

Name:	sendContactToPhone	
Aktor:	BlueSystem	
Zweck:	Kontakt an das Mobiltelephon senden	
Kurzbeschreibung:	Soll einen Kontakt an das Mobiltelephon übermitteln	
Priorität:	Primär	
Ablauf:	BlueSystem-Aktionen	Systemantwort
	1. Stellt anstehende Kontakt Übertragung fest	2. Übermittelt den Kontakt an das Mobiltelephon

Name:	sendSmsToPhone	
Aktor:	BlueSystem	
Zweck:	SMS an das Mobiltelephon senden	
Kurzbeschreibung:	Soll ein SMS an das Mobiltelephon übermitteln	
Priorität:	Primär	
Ablauf:	BlueSystem-Aktionen	Systemantwort
	1. Stellt anstehende SMS Übertragung fest	2. Übermittelt das SMS an das Mobiltelephon

Name:	readSettings	
Aktor:	BlueSystem	
Zweck:	Einlesen der Konfiguration	
Kurzbeschreibung:	Die Konfigurations Datei der Applikation soll gelesen und deren Werte der Applikation bekannt gemacht werden	
Priorität:	Primär	
Ablauf:	BlueSystem-Aktionen	Systemantwort
	1. Stellt geänderte Konfiguration fest	2. Liest Die Konfigurations Datei der Applikation ein und macht deren Werte der Applikation bekannt

Name:	setSettingsView	
Aktor:	BlueSystem	
Zweck:	Konfiguration anzeigen	
Kurzbeschreibung:	Eine vorhandene Konfiguration soll dem Benutzer im Tab Settings angezeigt werden	
Priorität:	Primär	
Ablauf:	BlueSystem-Aktionen	Systemantwort
	1. Stellt vorhandene Konfiguration fest	2. Die Konfiguration wird im Tab Settings angezeigt

Name:	showMetaInfo	
Aktor:	BlueSystem	
Zweck:	Zeigt Meta Informationen einer Datei	
Kurzbeschreibung:	Es sollen Metainformationen einer Datei wie Grösse, Typ und bei Bilddateien auch die Abmessungen des Bildes in Pixeln dem Benutzer angezeigt werden.	
Priorität:	Sekundär	
Ablauf:	BlueSystem-Aktionen	Systemantwort
	1. Stellt fest, dass eine Datei ausgewählt worden ist.	2. Sofern lesbar werden Metainformationen der Datei wie Grösse, Typ und bei Bilddateien auch die Abmessungen des Bildes in Pixeln angezeigt

5 Design

5.1 Entscheidungen

5.1.1 Programmiersprache

Als Programmiersprachen für eine Applikation unter Linux mit benutzerfreundlichem GUI kommen grundsätzlich in Frage:

- C++ (diverse Möglichkeiten für das GUI)
- Java
- TCL (mit TK für das GUI)
- Perl (mit GTK für das GUI)

Entscheidungsfindung

Alle vier Sprachen wurden in Betracht gezogen.

TCL und Perl sind früh ausgeschieden, da sich nicht um Hoch- sondern um Scriptsprachen handelt. Ich verfüge bei beiden über eine gewisse Erfahrung, aber eher zu wenig für eine Applikation im Umfang dieses Projektes. Weiter kenne ich keine Ansprechperson, die mir bei Problemen mit einer dieser beiden Sprachen hätte helfen können. Schlussendlich noch ein letzter Minus Punkt, eine freie Wahl der Graphikbibliothek ist bei beiden Sprachen theoretisch gegeben, in der Praxis mit vernüftigem Aufwand verwendbar ist aber nur TCL mit TK und Perl mit GTK. Leider existiert aber weder für TK noch für GTK eine Software, um das GUI mittels einer graphischen Oberfläche zu entwickeln, das GUI müsste im Quelltext mit einem Texteditor „gestaltet“ werden.

Java ist eine sehr interessante Option, vor allem wegen seiner Plattformunabhängigkeit und der vorhandenen, guten Entwicklungsumgebungen. Ich besitze leider gar keine Erfahrung bezüglich Entwicklung unter Java und kenne niemanden, der mir bei Problemen hätte weiterhelfen können. Trotzdem wagte ich mich an ein kleines Musterprojekt. Ich musste aber feststellen, dass ich sehr früh auf grundsätzliche Probleme stiess, deren Lösung mich zu viel Zeit gekostet hätte und deren Ursache lediglich meine sehr rudimentären Java Kentnisse sind. Somit war auch Java ausgeschieden.

So blieb nur das mir vertaute und an der TSU behandelte C++ als letzte Option.
C++ ist bezüglich Plattformunabhängigkeit ebenfalls sehr geeignet, sofern keine proprietären Bibliotheken wie die MFC verwendet werden. Weiter bin ich als Entwickler unter C++ völlig frei, was die Wahl der Graphikbiblithek angeht. Und last but not least, ich kenne Menschen, die ich bei Problemen mit C++ fragen kann.

Entscheidung

Meine Wahl der Programmiersparche für dieses Projekt viel auf C++ .

5.1.2 Bibliotheken

Die Wahl der zu verwenden Bibliotheken war, mit Ausnahme der Graphikbibliothek, gar keine, da keine Alternativen vorhanden sind.

Verwendete Bibliotheken

- STL
- Bluetooth (BlueZ)
- OpenOBEX
- ezV24
- QT
- KDE

Bemerkungen zu den verwendeten Bibliotheken

Die STL findet wohl für jede Applikation unter C++ Verwendung, so auch bei diesem Projekt.

Bluetooth (BlueZ) ist die Linux Implementation der Bluetooth Spezifikationen. Siehe Punkt 4.1.3 für weitere Erläuterungen zu Bluetooth.

OpenOBEX wird für den Dateiautausch zwischen PC und Mobiltelephon benötigt. Siehe Punkt 4.1.5 für weitere Erläuterungen zu OpenOBEX.

ezV24 ist nicht zwingend nötig, bedeutet für mich aber eine deutliche Vereinfachung des Enwicklungsprozesses. Siehe Punkt 4.1.4 für weitere Erläuterungen zu ezV24.

QT dient in diesem Projekt primär der Darstellung des GUI. Siehe Punkt 4.2.2 für weitere Erläuterungen zu QT.

KDE wird benötigt, da als Entwicklungumgebung KDevelop verwendet wird, siehe auch Punkt 5.1.3 und Punkt 4.2.4. KDE stellt das Applikationframework von BluePhone. Ich habe aber darauf geachtet, dass die Applikation mit geringem Aufwand umgeschrieben werde könnte, um die Abhängigkeit von KDE zu entfernen.

5.1.3 Entwicklungsumgebung

Als Entwicklungsumgebungen kommen in Frage:

- Kylix von Borland
- Anjuta
- KDevelop
- SNIFF+ von Wind River System
- Eclipse von IBM

Entscheidungsfindung

Alle fünf Entwicklungsumgebungen wurden in Betracht gezogen und ich habe mit allen ein kleines „Hello World“ Projekt erstellt.

Alle sind komplette, stabile IDEs, die ein C++ Projekt vewalten können.
Die Wahl viel trotzdem relativ schnell auf KDevelop, zugegebenermassen hauptsächlich aus einem einzigen, nicht technischem Grund.
KDevelop ist eine dem „Visual Studio“ von Microsoft sehr ähnliche Entwicklungsumgebung. Für mich ist die Nähe zur Bedienung von Visual Studio ein grosser Vorteil, da an der TSU der C++ Unterricht ausschlisslich unter Visual Studio erteilt wurde.

Entscheidung

Meine Wahl der Entwicklungsumgebung für dieses Projekt viel somit auf KDevelop.

5.1.4 Datenhaltung

Für die Datenhaltung kommen in Frage:

- Datenbank
- Binary Daten File
- Textfile

Entscheidungsfindung

Alle drei Möglichkeiten der Datenhaltung wurden in Betracht gezogen.

Aufgrund der kleinen Datenmenge, die ein Mobiltelephon speichern und liefern kann, macht eine Datenbank für dieses Projekt keinen Sinn, allein der Overhead des Datenbankfiles würde die Datenmenge überschreiten.
Weiter wird die Performance von BluePhone vom Mobiltelphon beschränkt. Mobiltelephone liefern naturgemäss nur einen Bruchteil der Leistung eines Pcs, auch eines älteren Modells. Somit spielen auch mögliche Performance Vorteile einer Datenbanklösung keine Rolle.

Da die Performance keine Rolle spielt, macht auch eine Lösung mit einem Binary Daten File keinen Sinn.

Weiter hat ein Textfile drei grosse Vorteile gegenüber den anderen beiden Lösungen. Erstens ist die Datenspeicherung in ein Textfile sehr einfach zu implementieren, zweitens kann ein Textfile auf jedem beliebigen Computer gelesen werden und drittens kann ein Textfile auch ohne Drittsoftware direkt vom Menschen gelesen werden. Letzteres kann zum Beispiel von Vorteil sein, wenn ein Mobiltelephon abhanden kommt und das neu erstandene Nachfolgermodell nicht kompatibel zu BluePhone ist. So könnte ein Backup der Kontake wenigstens noch von Hand wieder auf das neue Mobiltelephon übertragen werden.

Entscheidung

Meine Wahl der Datenhaltung für dieses Projekt viel somit auf das Textfile.

5.2 Klassendiagramm

5.2.1 Übersicht der Klassen

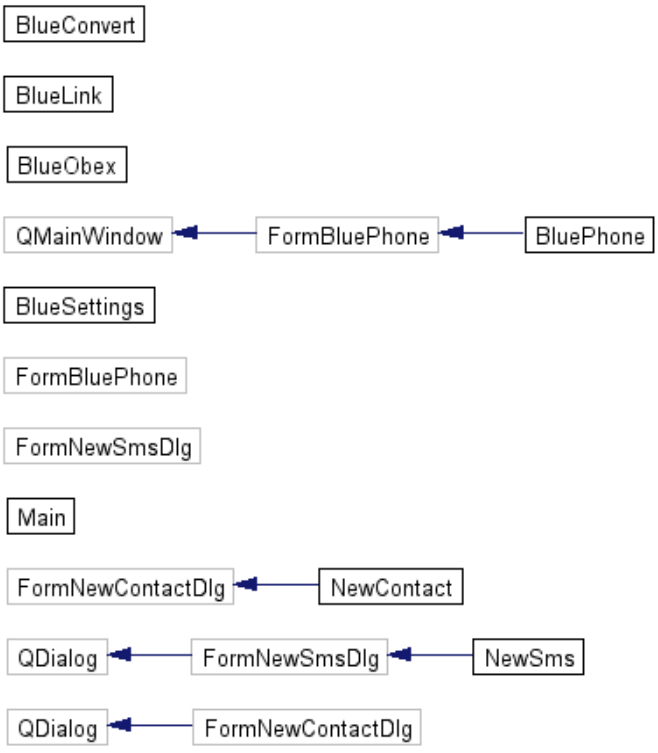
Klassenname	Funktionalität
BluePhone	Hauptklasse von BluePhone. Dies ist die Basisklasse von BluePhone, von ihr aus wird die ganze Applikation gesteuert. Diese Klasse enthaelt sämtliche Slots & Signals, die von QT-Designer während des UI-Designs erstellt wurden sowie einige zusätzliche Methoden. Sämtliche Use Cases werden von dieser Klasse behandelt und bei Bedarf zur Weiterverarbeitung an weitere Klassen übergeben.
BlueLink	Diese Klasse stellt die serielle Verbindung zum Telephon bereit. Als Träger werden Bluetooth, IrDA und serielle Kabel unterstützt.
BlueConvert	Konvertiert 7bit GSM Zeichensaetze vom Telephon und zurueck. Diese Klasse dient der Umwandlung der 7Bit GSM Zeichensätze vom Telephon zu PC Unicode und umgekehrt, weiter der Umwandlung des "ERICSSON" Zeichensatzes zu PC Unicode und ebenfalls auch umgekehrt. Sie stellt ebenfalls Methoden zur codierung oder decodierung eines SMS PUD zur Verfügung. Einige Methoden (v.a. die riesen Cases) dieser Klasse sind übernommen von scmxx, einer älteren Suite von Terminalprogrammen für ältere Simens Mobil Telephone. scmxx wurde von Hendrik Sattler geschrieben und unter GPL veröffentlicht. Die Programme sowie deren Source sind unter http://www.hendrik-sattler.de/scmxx/ verfügbar. Die übernommenen Methoden habe ich an meine Bedürfnisse angepasst.
BlueObex	Stellt OBEX zur Verfügung. Diese Klasse stellt Schnittstellen zu OBEX zur Verfügung und enthält alle Daten zu einer Obex Session. Die Struktur und die Methoden sind aus einer OpenOBEX Demo Applikation entstanden. Diese finden sich, in ähnlicher Form, in vielen Applikationen, die OpenOBEX verwenden.
BlueSettings	Speichert die Einstellungen von BluePhone. Diese Klasse speichert die Einstellungen von BluePhone unter Verwendung von KConfig ab. Das Konfigurationsfile liegt in ~/.kde/share/config/bluephonerc.

NewContact	Erstellt einen NewContactDlg. Verarbeitet die Eingaben des Benutzers und leitet diese an BluePhone weiter. Falls editiert wird, erwartet der Konstruktor die Daten des selektierten Eintrages.
NewSms	Erstellt einen NewSmsDlg. Verarbeitet die Eingaben des Benutzers und leitet diese an BluePhone weiter. Falls ein SMS beantwortet wird, erwartet der Konstruktor die Daten des selektierten SMS.

5.2.2 Definitives Klassendiagramm

Folgendes Klassendiagramm wurde anhand der Punkte, die in der Analyse und dem Design erarbeitet wurden geplant und schlussendlich von Doxygen aus dem BluePhone Quellcode generiert.

Die Klassen, die den Suffix Dlg aufweisen, sind mit QT Designer erstellt und dienen der Darstellung des GUIs respektive von Dialogen. Sie sind nicht dokumentiert, da sie aus einem QT UI File (XML) generiert werden und jede Dokumentation bei der kleinsten Änderung am GUI überschrieben werden würde.



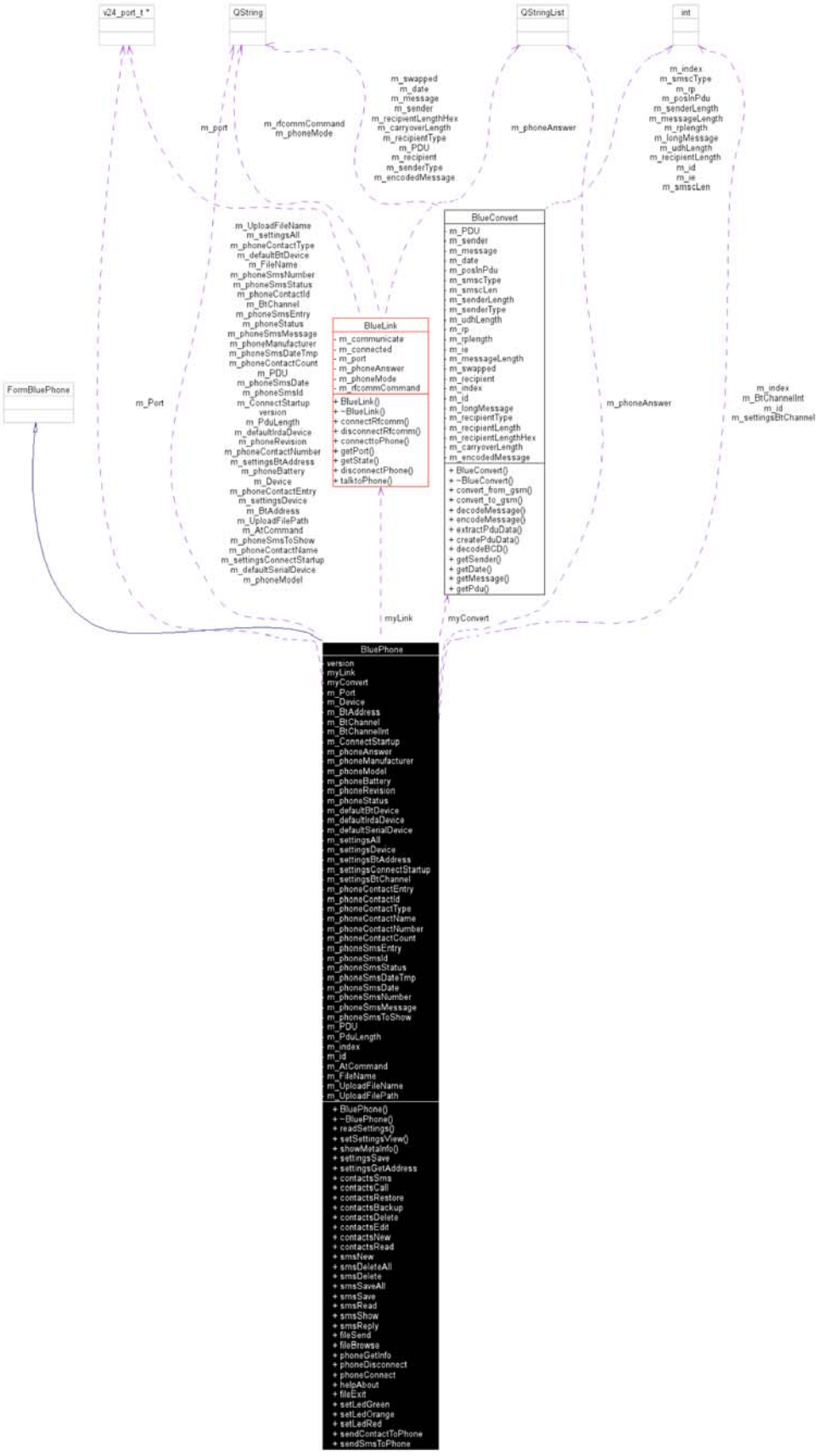
5.3 Kollaborationsdiagramme

5.3.1 Allgemein

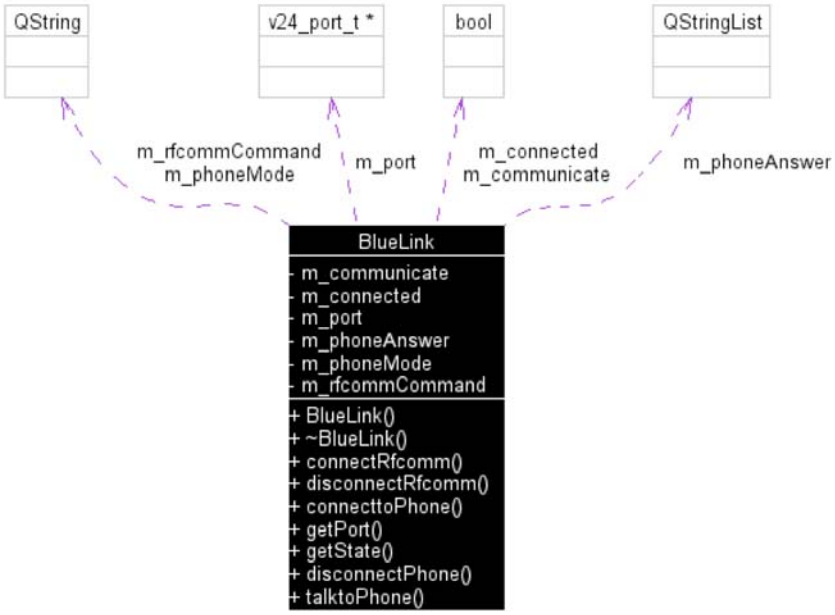
Die nachfolgend gezeigten Kollaborationsprogramme sind von Doxygen direkt aus dem Quellcode von BluePhone generiert worden und somit aktuell und gültig.

Die Lesbarkeit ist in diesem Dokument ist vor allem bei komplexeren Diagrammen beschränkt, es sei deshalb an dieser Stelle auf die sehr übersichtliche HTML Klassendokumentation auf der CD verwiesen.

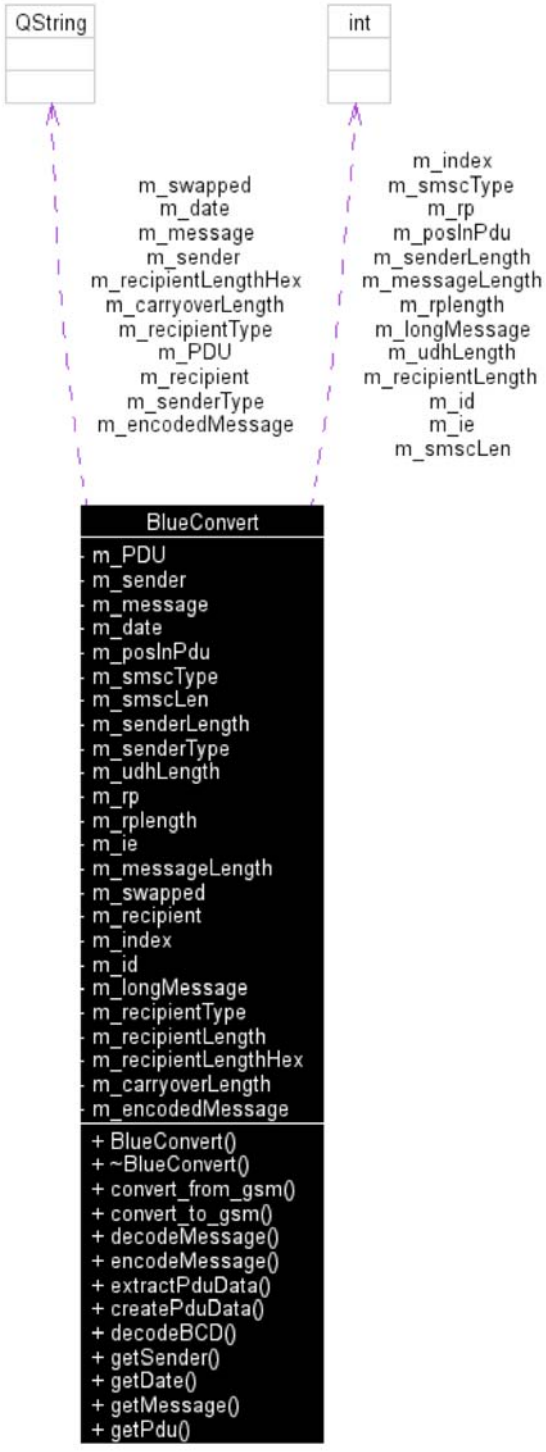
5.3.2 Kollaborationsdiagramm BluePhone



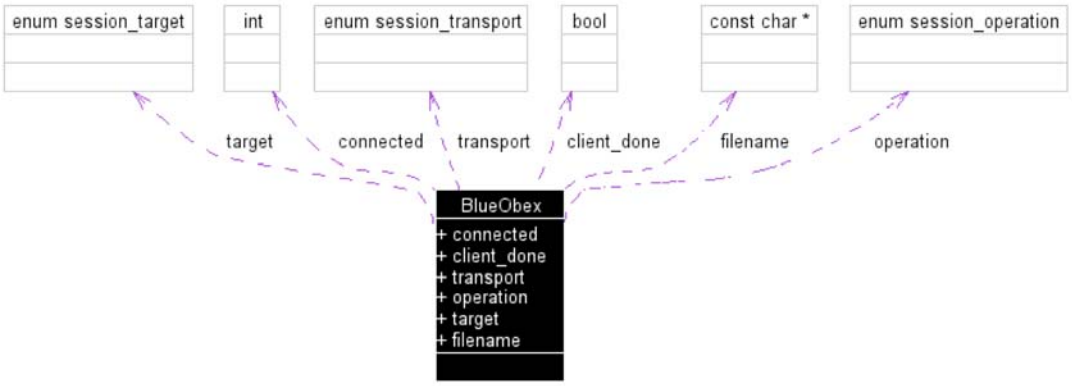
5.3.3 Kollaborationsdiagramm BlueLink



5.3.4 Kollaborationsdiagramm BlueConvert

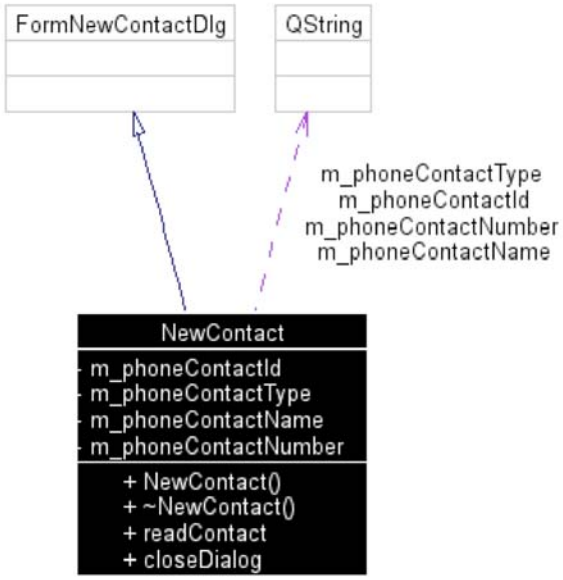


5.3.5 Kollaborationsdiagramm BlueObex

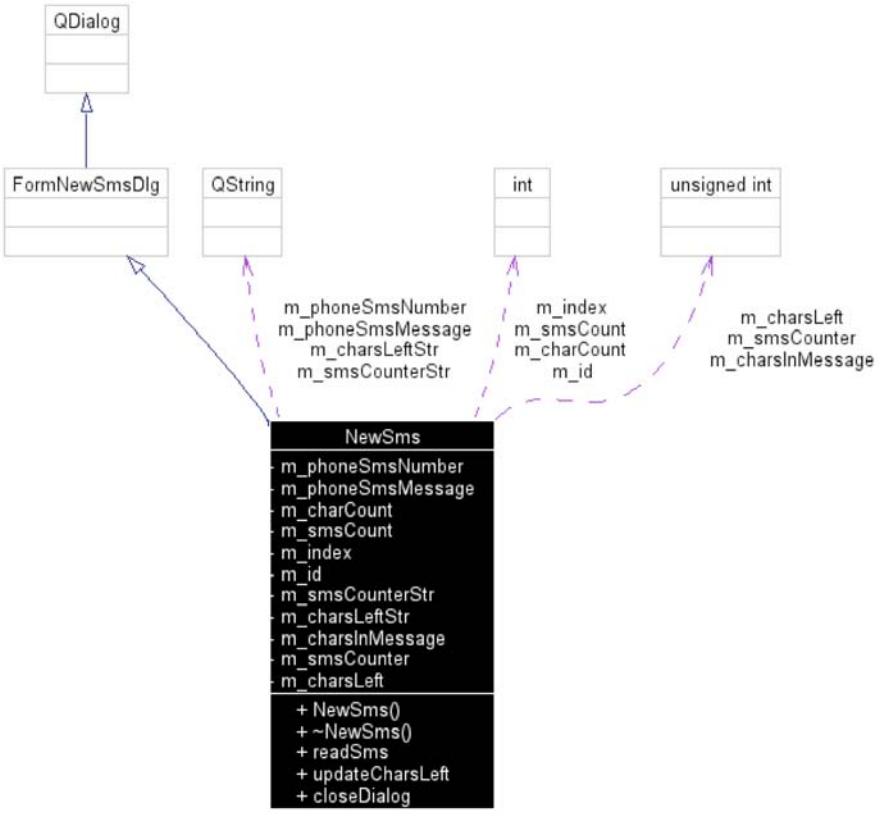


5.3.6 Kollaborationsdiagramm BlueSettings

5.3.7 Kollaborationsdiagramm NewContact



5.3.8 Kollaborationsdiagramm NewSms



5.4 GUI

5.4.1 Allgemein

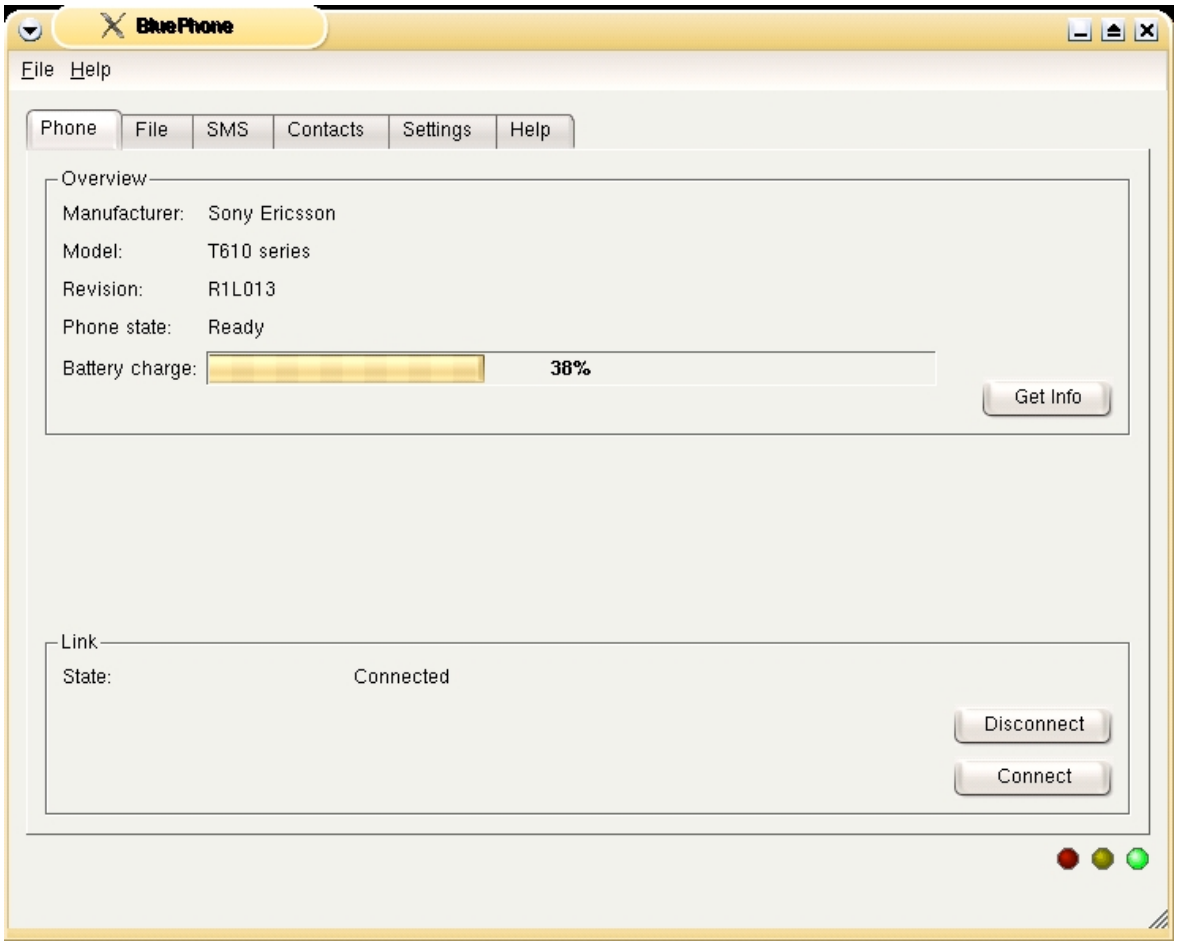
Das GUI von BluePhone soll gemäss Pflichtenheft benutzerfreundlich aber auch portabel sein. Dies wird, wie unter Punkt 4.2.2 und 5.1.2 beschrieben, mit QT erreicht.

Die Gestaltung des GUIs ist, wie unter Punkt 4.2.3 beschrieben, mittels des QT Designers realisiert.

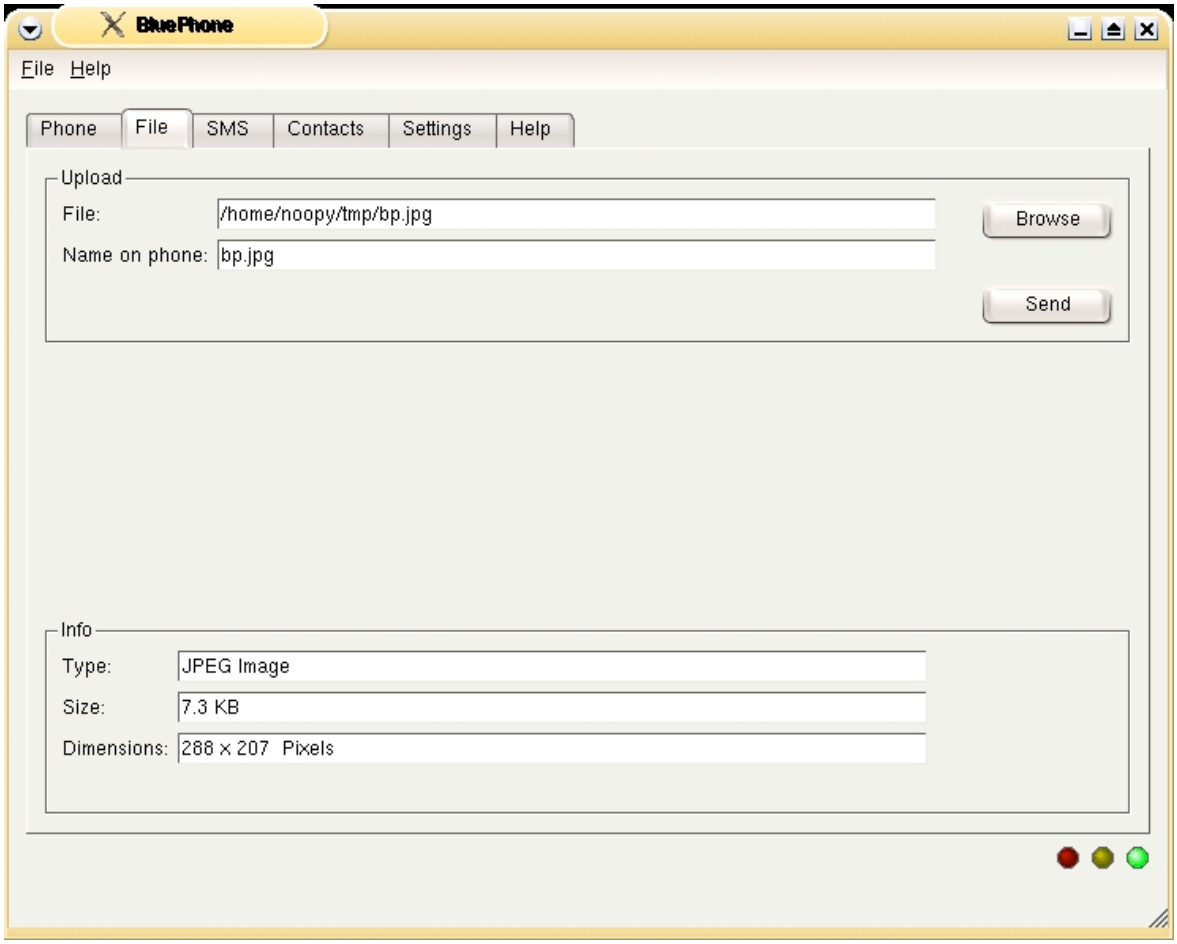
Es wurden verschiedene Möglichkeiten der Navigation getestet. Die Navigation über Tabs erwies sich letztendlich als die praktischste und wurde deshalb geplant und realisiert.

Das fehlende Icon in der Titelleiste ist übrigens auf einen Fehler der auf dem Entwicklungssystem verwendeten Version der KDE Bibliothek zurückzuführen. Dieses Phänomen tritt bei sämtlichen Programmen auf dem Entwicklungssystem auf. Tests auf Drittsystemen ergaben, dass das Problem mit einer neueren oder auch älteren Version der KDE Bibliothek verschwindet, die Software muss dazu nicht neu übersetzt werden.

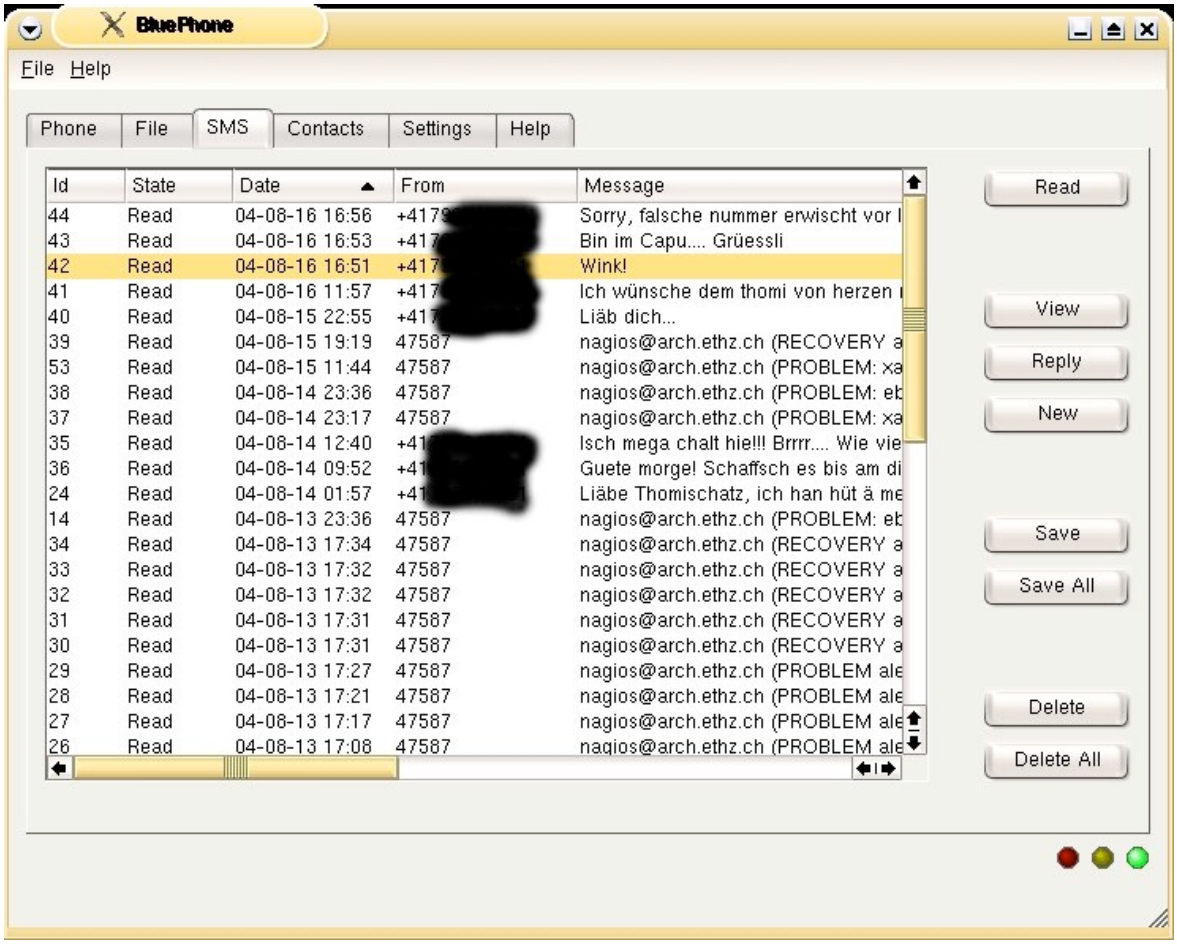
5.4.2 BluePhone "Phone" Tab



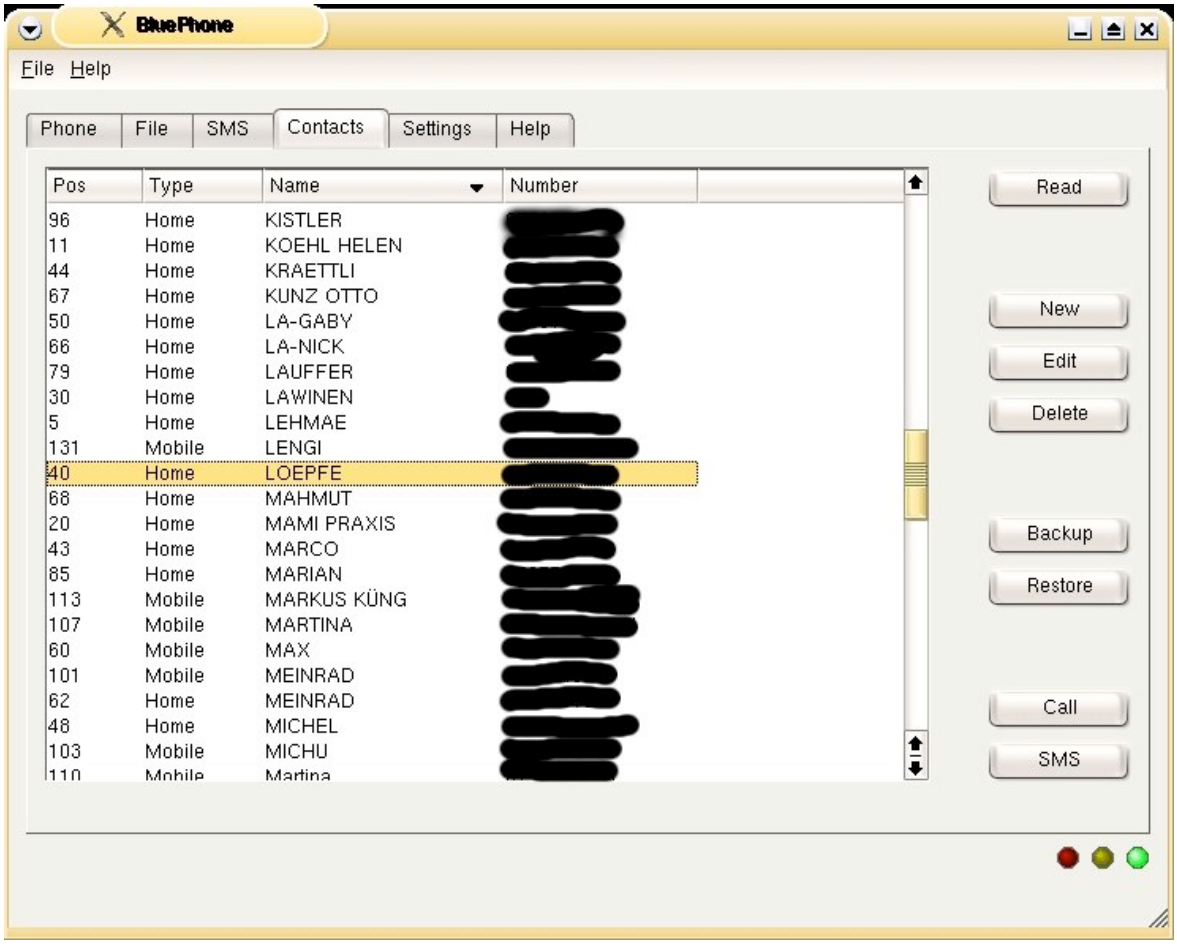
5.4.3 BluePhone “File” Tab



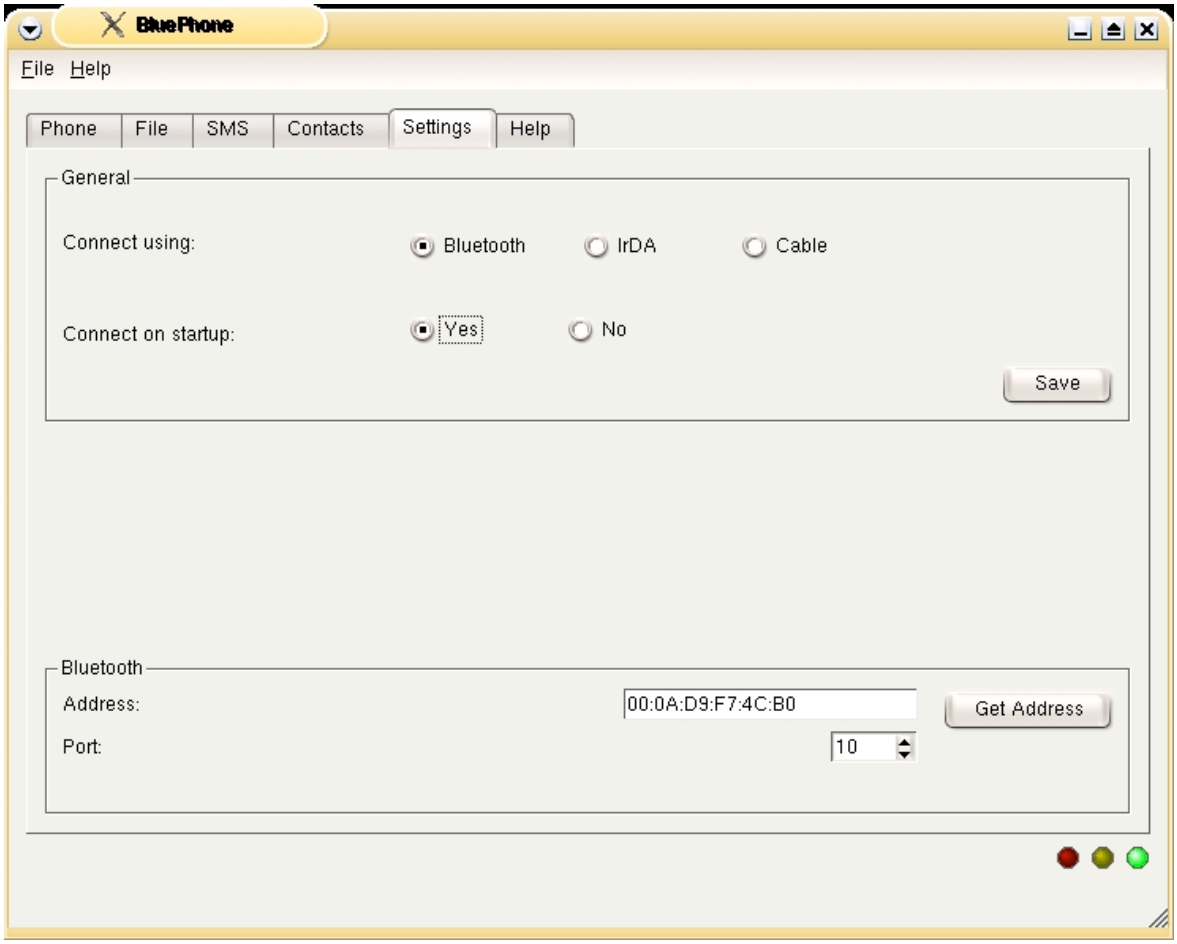
5.4.4 BluePhone "SMS" Tab



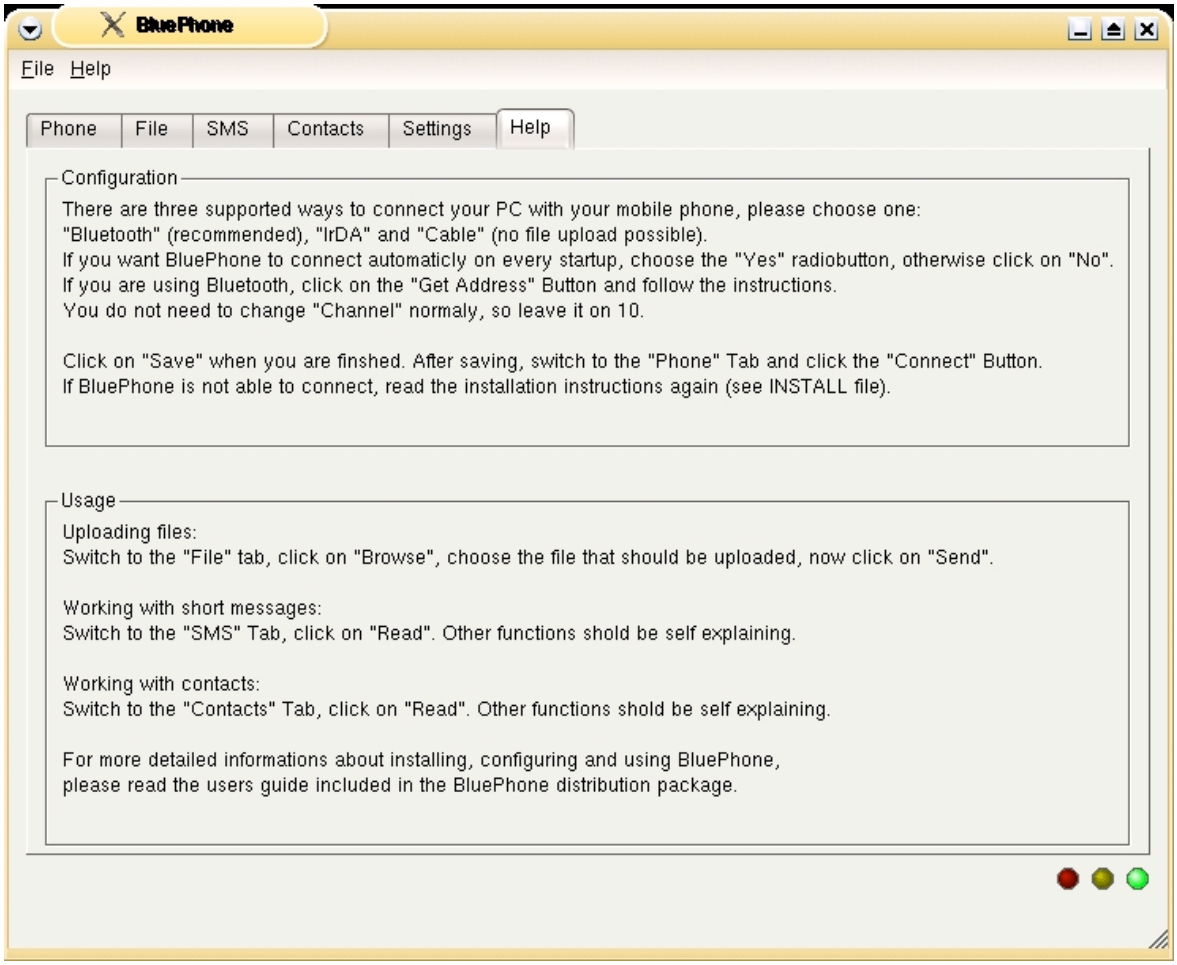
5.4.5 BluePhone “Contacts” Tab



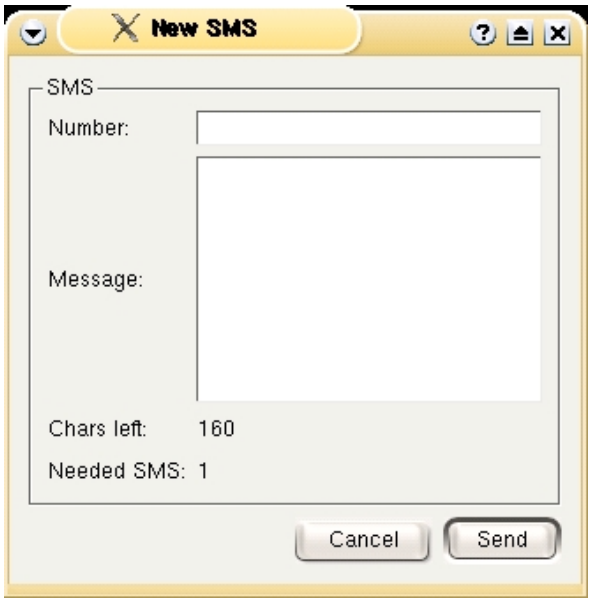
5.4.6 BluePhone “Settings” Tab



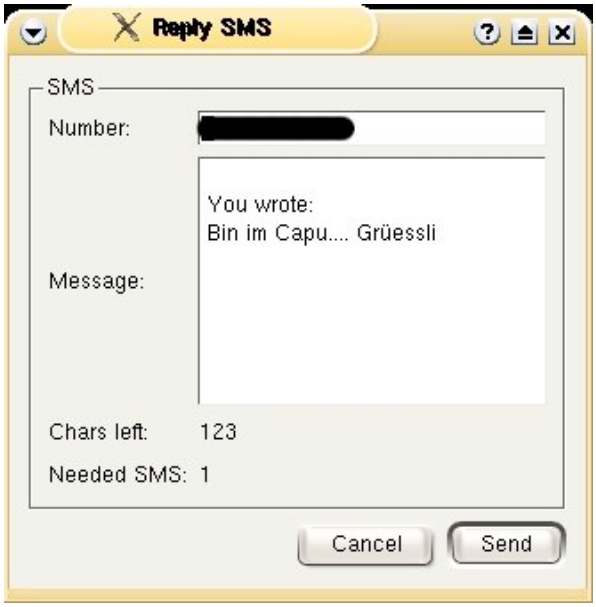
5.4.7 BluePhone “Help” Tab



5.4.8 BluePhone "New SMS" Dialog



5.4.9 BluePhone "Reply SMS" Dialog



5.4.10 BluePhone "New Contact" Dialog

5.4.11 BluePhone "Edit Contact" Dialog

6 Implementation

6.1 Verwendete Komponenten

Die verwendeten Komponenten sind unter Kapitel 4 und 5 detailliert aufgeführt.

Ich möchte an dieser Stelle wieder auf die Doxygen generierte HTML Dokumentation, die sich auf der CD befindet, verweisen. Mit Hilfe dieser sehr gut strukturierten, verlinkten Dokumentation kann sich der interessierte Leser einen guten Überblick über die von mir getätigte Implementation dieses Projektes verschaffen.

Weiter kann der Sourcecode so auf eine komfortable Weise betrachtet und studiert werden. Ich habe mich bemüht, den Quellcode ausführlich und korrekt zu dokumentieren, so dass ein Aussenstehender schnell die Funktionen und die Abläufe der Software verstehen kann.

6.2 Spezielles

6.2.1 Status LEDs

Die LEDs sollen Statusanzeige verdeutlichen, dass entweder

- Die Vebindung zum Mobiltelphon steht (grüne LED)
- Daten mit dem Mobiltelephon ausgetauscht werden (orange LED)
- Die Verbindung zum Mobiltelphon getrennt ist (rote LED)

Da Datentransfers sehr lange dauern können, würde sich der Benutzer ohne ein solches Feedback der Applikation fragen, ob die Applikation abgestürzt ist oder ob noch ein Prozess im Gange ist.

Die Implementation dieser Statusanzeige gestaltete sich als unerwartet schwierig und ist eigen.

Wenn bei einer Interaktion des Benutzers nur das Hauptfenster und keine Dialoge verwendet werden, geschieht das Setzen der LEDs folgendermassen:

- Der eine Methode aufrufende PushButton setzt auf das Ereigniss „pressed“ (nicht „clicked“!) die orange LED
- Die auf das Ereigniss „clicked“ aufgerufene Methode, setzt, nachdem sie ihre Aufgabe erfüllt hat, die grüne oder rote LED (je nach Status der Verbindung)

Dies kann leider nicht anders realisiert werden, da das GUI erst nach Beenden der Methode neu aufgebaut wird. Somit würde der Benutzer die orange LED gar nie leuchten sehen, wenn diese nicht kurz vor dem Aufrufen der eigentlichen Methode aktiviert würde.

Falls eine Methode einen Dialog erstellt, wird die orange LED kurz vor Aufbau des Dialoges von der aufrufenden Methode gesetzt.
Die grüne respektive rote LED wird vom Dialog nach Beenden seiner Aufgabe mittels eines QT Signals, das er an BluePhone sendet, aktiviert.

6.2.2 OpenOBEX

Ich benutze die OpenOBEX C Api.
Damit der Linker die Funktionen unterscheiden kann rsp. dass der cpp Compiler diese nicht irgendwie (um)benennt und so den Linker aus dem Tritt bringt, wird obex.h mittels extern "C" {} eingebunden.

7 Tests

7.1 Allgemeines

Um sicherzustellen, dass die Applikation keine Fehler mehr enthält, wurde Bluephone in drei Schritten getestet:

- Modultests während der Programmierung
- Anschlusstests durch den Programmierer
- Kontrolltests durch einen Benutzer

Hinweis, die Kontrolltests durch einen Benutzer werden in der Zeitspanne zwischen Abgabe der Dokumentation und Präsentation der Arbeit durchgeführt.

7.2 Modultests

Während der Implementation wurde jede Funktion und jede Klasse in sich selbst getestet. Für diese Tests wurde das „Black-Box" Verfahren angewendet. Das bedeutet, es wurden die Übergabe- und Rückgabe- Parameter einer Methode geprüft. Dieser Vorgang wurde bei jeder Funktion so lange wiederholt, bis das gewünschte Resultat dargestellt respektive übergeben worden ist.

Diese Tests wurden nicht protokolliert, da sie während der Implementationsphase gemacht worden sind und sich teilweise sehr oft mal wiederholt haben.

7.3 Abschlusstests

Eine Bemerkung zu den Abschlusstests. Für das das Ergebnis irrelevante Systemzustände werden in den Tabellen unter „Situation" nicht aufgeführt.

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation nicht gestartet Kein Configfile vorhanden	Applikation starten	Applikation startet Disconnected	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation nicht gestartet Configfile vorhanden	Applikation starten	Applikation startet mit den Parametern aus dem Configfile. Alle Konfigurations Parameter funktionieren wie erwartet	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected	Applikation via Menu File -> Exit beenden	Applikation trennt Verbindung und schliesst sich	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Disconnected	Applikation via Menu File -> Exit beenden	Applikation schliesst sich	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected.	Applikation via ☒ des Windowmanages beenden	Applikation schliesst sich	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Disconnected	Applikation via ☒ des Windowmanages beenden	Applikation schliesst sich	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet	Menu Help -> About aufrufen	About Messagebox erscheint	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Disconnected	pushButtonPhoneGetInfo klicken	Keine Reaktion	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Disconnected	pushButtonPhoneDisconnect klicken	Messagebox erscheint, dass das Telephon bereits getrennt ist	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Disconnected	pushButtonPhoneConnect klicken	Verbindung wird aufgebaut, grüne LED leuchtet auf, rote LED erlischt, Telephonstatus Informationen werden korrekt angezeigt	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected	pushButtonPhoneConnect klicken	Messagebox erscheint, dass das Telephon bereits verbunden ist	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected	pushButtonPhoneDisconnect klicken	Grüne LED erlischt, Verbindung wird getrennt, rote LED leuchtet auf	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected	pushButtonPhoneGetInfo klicken	orange LED leuchtet auf, Telephonstatus Informationen werden korrekt aufgefrischt, grüne LED leuchtet auf	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet	pushButtonFileBrowse klicken	File kann ausgewählt werden, nach der Selektion werden die Informationsfelder korrekt abgefüllt	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Disconnected	pushButtonFileSend klicken	Messagebox erscheint, dass das Telephon getrennt ist	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected	pushButtonFileSend klicken	Orange LED leuchtet, File wird auf das Telephon hochgeladen, grüne Led leuchtet wieder	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Disconnected	Sämtliche pushButton des SMS Tabs klicken	Messagebox erscheint, dass das Telephon getrennt ist	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste nicht gelesen	pushButtonSmsRead klicken	Orange LED leuchtet, SMS Liste des Telephons wird geladen, grüne Led leuchtet wieder	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste nicht gelesen	pushButtonSmsView klicken	Messagebox erscheint, dass erst eine Nachricht selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste nicht gelesen	pushButtonSmsReply klicken	Messagebox erscheint, dass erst eine Nachricht selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste nicht gelesen	pushButtonSmsSave klicken	Messagebox erscheint, dass erst eine Nachricht selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste nicht gelesen	pushButtonSmsSaveAll klicken	Messagebox erscheint, dass erst die SMS Liste gelesen werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste nicht gelesen	pushButtonSmsDelete klicken	Messagebox erscheint, dass erst eine Nachricht selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste nicht gelesen	pushButtonSmsDeleteAll klicken	Messagebox erscheint, dass erst die SMS Liste gelesen werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected	pushButtonSmsNew klicken	New SMS Dialog erscheint	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen	pushButtonSmsRead klicken	Orange LED leuchtet, SMS Liste des Telephons wird erneut geladen, grüne Led leuchtet wieder	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen, kein SMS selektiert	pushButtonSmsView klicken	Messagebox erscheint, dass erst eine Nachricht selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen, kein SMS selektiert	pushButtonSmsReply klicken	Messagebox erscheint, dass erst eine Nachricht selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen, kein SMS selektiert	pushButtonSmsSave klicken	Messagebox erscheint, dass erst eine Nachricht selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen, kein SMS selektiert	pushButtonSmsDelete klicken	Messagebox erscheint, dass erst eine Nachricht selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen	pushButtonSmsSaveAll klicken	File Save As Dialog erscheint, Zieldatei kann angegeben werden, SMS Liste wird korrekt in das Textfile abgespeichert	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen	pushButtonSmsDeleteAll klicken	Abfrage MessageBox erscheint, ob der Benutzer wirklich alle Nachrichten löschen will Bei Click auf No wird abgebrochen, bei Klick auf Yes leuchtet die orange LED, alle SMS auf dem Telephon werden gelöscht, die SMS Liste wird erneut eingelesen, die grüne LED leuchtet wieder.	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen, SMS selektiert	pushButtonSmsView klicken	Messagebox mit Nachricht als Inhalt erscheint	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen, SMS selektiert	pushButtonSmsReply klicken	Die orange LED leuchtet, ein Reply SMS Dialog mit den korrekten Werten des selektierten SMS wird geöffnet	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen, SMS selektiert	pushButtonSmsSave klicken	File Save As Dialog erscheint, Zieldatei kann angegeben werden, das SMS wird korrekt in das Textfile abgespeichert	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, SMS Liste gelesen, SMS selektiert	pushButtonSmsDelete klicken	Abfrage MessageBox erscheint, ob der Benutzer die Nachricht wirklich löschen will Bei Click auf No wird abgebrochen, bei Klick auf Yes leuchtet die orange LED, die selektierte SMS auf dem Telephon wird gelöscht, die SMS Liste wird erneut eingelesen, die grüne LED leuchtet wieder.	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste nicht gelesen	pushButtonContactsRead klicken	Orange LED leuchtet, Kontakte Liste des Telephons wird geladen, grüne Led leuchtet wieder	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste nicht gelesen	pushButtonContactsEdit klicken	Messagebox erscheint, dass erst ein Kontakt selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste nicht gelesen	pushButtonContactsDelete klicken	Messagebox erscheint, dass erst ein Kontakt selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste nicht gelesen	pushButtonContactsCall klicken	Messagebox erscheint, dass erst ein Kontakt selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste nicht gelesen	pushButtonContactsSms klicken	Messagebox erscheint, dass erst ein Kontakt selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste nicht gelesen	pushButtonContactsBackup klicken	Messagebox erscheint, dass erst die Kontakte Liste gelesen werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected	pushButtonContactsRestore klicken	Orange LED leuchtet, File Open Dialog erscheint, ein BackupFile kann selektiert werden, MessageBox mit Warnung erscheint, ob der Benutzer wirklich all seine Kontakte auf dem Telephon mit dem selektierten Backup File überspielen will, falls No geklickt wird Abgebrochen, falls Yes geklickt wird das Telephonbuch des Telephons erfolgreich überschrieben, die Kontakt Liste wird erneut eingelesen, die grüne LED leuchtet wieder	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste gelesen, kein Kontakt selektiert	pushButtonContactsEdit klicken	Messagebox erscheint, dass erst ein Kontakt selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste gelesen, kein Kontakt selektiert	pushButtonContactsDelete klicken	Messagebox erscheint, dass erst ein Kontakt selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste gelesen, kein Kontakt selektiert	pushButtonContactsCall klicken	Messagebox erscheint, dass erst ein Kontakt selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste gelesen, kein Kontakt selektiert	pushButtonContactsSms klicken	Messagebox erscheint, dass erst ein Kontakt selektiert werden muss	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste gelesen, Kontakt selektiert	pushButtonContactsEdit klicken	Orange LED leuchtet, Edit Contact Dialog mit den korrekten Werten des selektierten Kontaktes erscheint	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste gelesen, Kontakt selektiert	pushButtonContactsDelete klicken	Orange LED leuchtet, MessageBox erscheint, ob der Benutzer den selektierten wirklich löschen will, falls No wird abgebrochen, falls Yes wird der selektierte Kontakt erfolgreich gelöscht und die Kontakteliste erneut eingelesen, grüne LED leuchtet	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste gelesen, Kontakt selektiert	pushButtonContactsCall klicken	Der selektierte Kontakt wird angerufen	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, Kontakte Liste gelesen, Kontakt selektiert	pushButtonContactsSms klicken	Orange LED leuchtet, New SMS Dialog mit selektiertem Kontakt als Empfänger wird korrekt erstellt	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet	pushButtonSettingsSave klicken	Die gesetzte Konfiguration wird korrekt gespeichert	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet	pushButtonSettingsGetAddress klicken	MessageBox mit Anleitung zum Ausfindigmachen der BTADR erscheint	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet	Help lesen	Die Hilfe wird korrekt angezeigt	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, New oder Reply SMS Dialog aktiv	SMS senden	Falls Cancel geklickt wird abgebrochen, Falls Send geklickt wird das SMS korrekt verschickt, die grüne LED leuchtet wieder	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, New oder Reply SMS Dialog aktiv	Überlanges SMS senden	Falls Cancel geklickt wird abgebrochen, Falls Send geklickt werden die SMS korrekt verschickt, die grüne LED leuchtet wieder	✔

Situation	Tätigkeit	Ergebnis	Erfüllt
Applikation gestartet und Connected, New oder Edit Contact Dialog aktiv	Kontakt editieren	Falls Cancel geklickt wird abgebrochen, Falls Save geklickt wird der Kontakt korrekt gespeichert, die Kontakteliste wird erneut eingelesen, die grüne LED leuchtet wieder	✔

7.4 Stabilitätstest

Die Applikation selbst hat, soweit dies festgestellt werden konnte, keine Stabilitäts Probleme.

Es können aber, unter gewissen Umständen, Probleme mit der Verbindung zum Mobiltelephon entstehen. Siehe auch Punkt 7.5 für weitere Informationen diesbezüglich.

7.5 Auswertung der Testergebnisse

Die Applikation arbeitet auf Anhieb korrekt und wie geplant. Dies ist auf eine saubere Analyse, ein gutes Design und stetige Modultests während der Implementation zurückzuführen.

Die Stabilität der Software ist gut, einzig die Verbindung zum Mobiltelephon kann durch schnelles Trennen und wieder Verbinden gestört werden. Es kann auch Probleme beim Wiederverbinden des Mobiltelephons geben, wenn das Mobiltelephon in verbundenem Modus ausser Reichweite des PC's bewegt wird.

Das Schlechteste, was der Author in seinen Tests produzieren konnte, war dass das Mobiltelephon nicht mehr auf Befehle über Bluetooth reagierte und neu gestartet werden musste. Provoziert habe ich diese Situation durch gleichzeitiges Absetzen von Befehlen von zwei Rechnern aus.

Diese Probleme sind, mindestens zu einem Teil, auf das Mobiltelephon zurückzuführen. Die beobachteten Phänomene konnten auch mit professioneller, kommerzieller Software unter Windows XP und Mac OS X provoziert werden.

8 Bedienungsanleitung

8.1 Abhängigkeiten

BluePhone erfordert folgende Komponeten, die auf Ihrem PC installiert und konfiguriert sein müssen. Das Vorgehen der Installation dieser Komponeten ist abhängig von der verwendeten Distribution. Bitte konsultieren Sie im Zweifelsfalle die Dokumentation Ihres Linux Distributors.

- Kernel mit aktivierter Bluetooth Unterstützung
- KDE
- OpenOBEX Bibliothek
- ezV24 Bibliothek
- Bluetooth Blibliothek
- Bluetooth Tools

Optional:

- Kernel mit aktivierter IrDA Unterstützung
- IrDA Tools

8.2 Installation

Es existieren grundsätzlich zwei verschiedene Wege, BluePhone zu installieren.

- Binary Packet installieren
- Quell Text kompilieren

Hinweis, zum Kompilieren des Quelltextes brauchen Sie von allen obenstehend aufgeführten Abhängigkeiten zusätzlich noch deren Entwicklerbibliotheken.

Binary Packet installieren:

- Doppelklicken Sie im Verzeichnis „packages“ auf der CD entweder auf bluephone.dpkg, bluephone.tar.gz oder auf bluephone.rpm, je nach verwendetem Packetsystem Ihrer Distribution.
- Folgen Sie den Anweisungen des Installers

Kompilieren des Source Codes:

- Entpacken Sie die Datei bluephone.tar.gz in ein Verzeichnis Ihrer Wahl
- Öffnen Sie eine Unix Shell
- Wechseln Sie in das Verzeichnis, in das Sie BluePhone entpackt haben
- Geben Sie `./configure` ein
- Geben Sie `make` ein
- Geben Sie `su –c make install` ein

Nacharbeiten

Bei beiden Verfahren ist es bei einigen Distribution nötig, das Programm rfcomm mit einem SetUID Bit zu versehen, um einem normalen Benutzer genügend Rechte zu geben, eine RFCOMM Verbindung aufzubauen.

Gehen Sie dazu wie folgt vor:

- Öffnen Sie eine Unix Shell
- Geben sie `su –c chmod +s /usr/bin/rfcomm` ein

Nun müssen das Mobiltelephon und der PC noch „gepairt“ werden. Starten Sie diesen Vorgang am besten von Ihrem Mobiltelephon aus. Konsultieren Sie bei Unklarheiten bitte das Handbuch Ihres Mobiltelphons

8.3 Konfiguration

Nach dem ersten Start von BluePhone müssen Sie dieses erst konfigurieren.

Wechseln Sie dazu ins „Settings“ Tab und setzen Sie folgende Optionen:

- Art der benutzten Verbindung. Diese kann entweder „Bluetooth“, „IrDA“ oder „Cable“ sein. Es wird „Bluetooth“ empfohlen.
- Falls sie bei einem Programmstart von Bluephone wünschen, dass sich dieses automatisch mit Ihrem Mobiltelephon verbinden, klicken Sie „Connect on Startup“ „Yes“, andernfalls klicken Sie auf „No“.
- Falls Sie Bluetooth benutzen, klicken Sie den Button „Get Address“ und folgen Sie den Anweisungen.
- „Channel“ ist normalerweise „10“.
- Klicken Sie auf „Save“.
- Wechseln Sie ins „Phone“ Tab und klicken Sie „Connect“.

Falls die Verbindung fehl schlägt, überprüfen Sie bitte, ob Ihre Linux Installation alle Erfordernisse von BluePhone erfüllt und ob BluePhone korrekt installiert ist. Konsultieren Sie im Zweifelsfalle die Dokumentation Ihres Linux Distributors.

Hinweis, bei Verwendung von „Cable“ als Verbindung entfällt die Möglichkeit, Dateien auf das Telephon hochzuladen.

8.4 Dateien hochladen

Um eine Datei auf das Mobiltelephon hochzuladen gehen Sie wie folgt vor:

- Wechseln Sie ins „File“ Tab.
- Klicken Sie auf „Browse“ und wählen Sie die Datei aus, die Sie auf das Mobiltelephon hochladen möchten.
- Klicken Sie auf „Send“.

Hinweis, Ihr Mobiltelephon unterstützt nur eine begrenzte Anzahl Datei Formate, verfügt im Vergleich zu Ihrem Computer über sehr wenig Speicherplatz und einen sehr kleinen Display.

Konsultieren Sie im Zweifelsfall bitte das Handbuch Ihres Mobiltelphons und beachten Sie die Anzeigen der „Info“ Felder in BluePhone, nachdem Sie eine Datei selektiert haben.

8.5 SMS Dienste nutzen

Wechseln Sie ins „SMS“ Tab und klicken Sie auf „Read“.

Sie haben nun folgende Dienste zur Verfügung:

SMS lesen
- Selektieren Sie eine Nachricht und klicken Sie auf „View“ oder doppelklicken Sie eine Nachricht.

SMS beantworten
- Selektieren Sie eine Nachricht und klicken Sie auf „Reply“.

Neue SMS erstellen
- Klicken Sie auf „New“.

Eine SMS sichern
- Selektieren Sie eine Nachricht und klicken Sie auf „Save“.
- Wählen Sie den Ort und den Namen der Datei, in der Sie das SMS sichern möchten.
- Klicken Sie auf „Ok“.

Alle SMS sichern
- Klicken Sie auf „Save All“.
- Wählen Sie den Ort und den Namen der Datei, in der Sie alle SMS sichern möchten.
- Klicken Sie auf „Ok“.

Eine SMS löschen
- Selektieren Sie eine Nachricht und klicken Sie auf „Delete“.
- Klicken Sie auf „Yes“, wenn Sie die Nachricht wirklich löschen möchten.

Alle SMS löschen
- Klicken Sie auf „Delete All“.
- Klicken Sie auf „Yes“, wenn Sie wirklich alle Nachrichten löschen möchten.

8.6 Kontakte Dienste nutzen

Wechseln Sie ins „Contacts“ Tab und klicken Sie auf „Read“.

Sie haben nun folgende Dienste zur Verfügung:

Neuen Kontakt erstellen
- Klicken Sie auf „New“.

Kontakt editieren
- Selektieren Sie einen Kontakt und klicken Sie „Edit“ oder doppelklicken Sie einen Kontakt.

Kontakt löschen
- Selektieren Sie einen Kontakt und klicken Sie auf „Delete“.
- Klicken Sie auf „Yes“, wenn Sie den Kontakt wirklich löschen wollen.

Alle Kontakte sichern
- Klicken Sie auf „Backup“
- Wählen Sie den Ort und den Namen der Datei, in der Sie alle Kontakte sichern möchten.
- Klicken Sie auf „Ok“.

Alle Kontakte auf dem Mobiltelephon wieder herstellen
- Klicken Sie auf „Restore“.
- Wählen Sie eine Datei aus, in der Sie zu einem vorhergehenden Zeitpunkt alle Kontakte gesichert haben.
- Klicken Sie auf Ok.

Warnung, diese Aktion löscht alle bestehenden Kontakte auf Ihrem Mobiltelephon!

Kontakt anrufen
- Selektieren Sie einen Kontakt und klicken Sie „Call“.

SMS an Kontakt verschicken
- Selektieren Sie einen Kontakt und klicken Sie „SMS“.

8.7 Diverses

Bitte beachten Sie die drei LEDs am rechten, unteren Rand von BluePhone.

Diese LEDs haben folgende Bedeutung:

- Grün: Das Mobiltelephon ist mit dem PC verbunden und wartet auf Anweisungen
- Orange: Der PC und das Mobiltelphon tauschen Daten aus, bitte haben Sie etwas Geduld
- Rot: Die Verbindung vom Mobiltelephon zum PC ist getrennt

Wenn Sie die Dienste von Bluephone länger nicht benötigen, trennen Sie bitte die Verbindung. Eine hergestellte Verbindung belastet den Akku Ihres Mobiltelephones sehr stark.

Gehen Sie dazu wie folgt vor:

- Wechseln Sie ins „Phone“ Tab
- Klicken Sie auf „Disconnect“

Falls Sie den aktuellen Stand der Akkuladung Ihres Mobiltelephones erfahren möchten, gehen Sie wie folgt vor:

- Wechseln Sie ins „Phone“ Tab
- Klicken Sie auf „Get Info“

9 Erfahrungsberichte und Erkenntnisse

9.1 Negatives

Ich war überrascht, wie sehr ich den Umfang der Aufgabe, den ich mir mit dieser Diplomarbeit gestellt gabe, unterschätzt habe.

Die Einarbeitung in die Themen Softwareentwicklung unter Linux sowie Datenaustausch mit mobilen Geräten unter Linux haben deutlich mehr Zeit als, bei Abgabe der Aufgabenstellung ursprünglich geplant, benötigt. Diese Abklärungen haben einen Grossteil der Zeit, die für die Analyse aufgewendet wurde, benötigt.

Weiter unterschätzte ich den Umstieg der Entwicklungsumgebung.
Der Wechsel von Microsoft Visual Studio unter Verwendung der MFC auf KDevelop unter Verwendung von STL/QT kostete mich einiges an Lehrgeld. Ich arbeitete insgesamt vier Tutorials durch, um mir eine fundierte Wissensbasis bezüglich der Vorgehensweise der Software Entwicklung mit diesen Werkzeugen zu schaffen.
Die dazu benötigte Zeit ist in dieser Dokumentation nicht aufgeführt. Mein virtueller Arbeitgeber, die „Me Myself and I AG“ setzte diese Kenntnisse bei Vergabe des Auftrages voraus, weshalb ich mir dieses Wissen in meiner Freizeit aneignete.

Diese Punkte werden mir bezüglich Aufwandschätzungen für die Zukunft eine Lehre sein.

9.2 Positives

Ich bin sehr erfreut über die Tatsache, dass ich mein erstes Software Projekt komplett selbstständig und erfolgreich habe durchführen können. Vor allem da die Terminsituation zeitweise sehr kritisch war.

Weiter habe ich an all den Punkten, die obenstehend unter Negatives aufgeführt sind, sehr viel gelernt. Der Mensch lernt am besten aus Fehlern.

Auch werden mich die Erfahrungen, die ich während allen Phasen dieses Projektes bezüglich Projektmanagement gemacht habe, in Zukunft begleiten.

Last but not least freut es mich, dass ich endlich einmal eine Applikation geschrieben habe, die ich auch in Realität verwenden kann. Einer Erweiterung von BluePhone steht nichts im Weg, die Basis steht. Die 222 Seiten starke AT Referenz spezifizert noch viele praktische Befehle, deren Implementation nur noch wenig Aufwand benötigt. Ich bekam bei der Implementation richtig Freude an meiner Applikation und musste mich teilweise selber bremsen, um den Blick auf das Wesentliche nicht zu verlieren.

9.3 Fazit

Die Applikation erfüllt zwei „Muss“ Punkte des Pflichtenheftes nicht. Diese sind:

Datenaustausch
- herunterladen Bilder
- herunterladen Töne

Kalender
- sichern

Beiden Punkten liegt ein OpenOBEX „ Get Data“ zugrunde. Das Herunterladen von Dateien vom Mobiltelephon konnte aus zeitlichen Gründen nicht mehr implementiert werden. Einerseits liegt die Ursache dieser fehlenden Zeit im obenstehend angesprochenen Unterschätzen der „Problem Domain“, andererseits auch daran, dass ich für meinen Arbeitgeber alle Server mit diversen Diensten wie Mail, File, Web, Groupware, etc. für 500 Angestelle und 2000 Studenten betreue.
Diese Server und deren Dienste werden per Mitte September auf neue Hard- und Software migriert, was primär meine Aufgabe ist.
Dieses paralelle Projekt ist mindestens gleich aufwendig wie diese Diplomarbeit und brachte mein Zeitmanagement ins Trudeln und stellte meine Belastbarkeit auf eine Probe.

So musste ich während der Analyse Prioritäten setzen und das Nichtimplementieren dieser Funktionen in Kauf nehmen.

Andererseits konnte ein Punkt aus dem „Soll“ Teil des Pflichtenheftes implementiert werden. Dieser ist:

Kommunikation
- Serielles Kabel
- USB Kabel
- IrDA

Während der Analyse zeigte sich, dass alle Schnittstellen von BluePhone als serielles Gerät angesprochen werden und somit gleich behandelt werden können. Dies wiederum ermöglichte die einfache Erweiterung der Kommunikationsmöglichkeiten um oben genannte Punkte.
An dieser Stelle sei hinzugefügt, das die Kommunikation über Kabel ungetestet ist. Dies hat finanzielle Gründe. Ein dazu benötigtes Kabel würde mehr kosten, als ich für mein Mobiltelephon ausgegeben habe.

9.4 Ausblick

Da ich mittlerweile, aus geschäftlichen Gründen, hauptsächlich unter Mac OS X arbeite, werde ich BluePhone mit der vielen Freizeit, die ich nach Abschluss meines ETH Projektes und der TSU haben werde, auf Mac OS X portieren.

Der Programmiervirus hat mich erfasst, so wie es mir unser erster C++ Lehrer, Markus Burri, im dritten Semester prophezeit hatte.

10 Unterschriften

10.1 Betreuer

Ich, Kurt Burri, bestätige, die Bestimmungen durchgelesen und den Diplomanden zu einer selbständigen Arbeit angeleitet und betreut zu haben.

Datum: ..... Unterschrift: .....

10.2 Diplomand

Totale Anzahl benötigter Arbeitsstunden für die Diplomarbeit: 225

Ich, Thomas Gemperli, bestätige, nach den unter Punkt 2 aufgeführten Bestimmungen gearbeitet und die Arbeit selbständig durchgeführt zu haben.

Datum: ..... Unterschrift: .....

11 Anhang

11.1 Begriffserklärungen

AT
AT ist eine de facto Standard Sprache zur Kontrolle von Modems. Moderne Mobiltelephone sind auch ein Modem.

Bluetooth
Ein Standard, der kabellose Kommunikation über Funk Signale definiert

BlueZ
Die Implementation des Bluetooth Protokolls des Linux Kernels.

ETSI
European Telecommunications Standards Institute
Gibt in Europa u.v.a den GSM Standard frei.

gcc
Der GNU C und C++ Compiler
Siehe auch: http://savannah.gnu.org/projects/gcc/

GNU
Die GNU Free Softare Foundation setzt sich für freie Software ein und entwicklt u.a. den gcc und g++
Siehe auch: http://www.gnu.org und http://savannah.gnu.org

GPL
Software Lizenz, die häufig für Open Source Software Projekte verwendet wird. Sie schützt ein "Software Werk" urheberrechtlich und besagt, dass der Quellcode der Software von jedem weiter verwendet werden darf, auch für eine kommerzielle Nutzung. Bedingung ist, das der verwendete Code weiterhin im Quelltext verfügbar bleibt und das der Name des Authors (Urheberrechtsinhaber) dort erwähnt bleibt.
Siehe auch: http://www.gnu.org/copyleft/gpl.html

GSM
Global System for Mobile communications.
Ein Standard des GSM Association, der die Schnittstellen von Mobiltelephonen definiert.
Siehe auch: http://www.gsmworld.com

IrDA
Ein Standard, der kabellose Kommunikation über Infrarot Signale definiert
Eine Organisation, die IrDA und Bluetooth spezifiziert hat

Linux
Der Kernel des umgangssprachlich als Linux bezeichneten Betriebsystems.
Siehe auch: http://www.kernel.org und http://www.linux.org

Natel
Schweizerischer, von Swisscom geschützter, Begriff für ein Mobiltelephon.

OpenOBEX
Die freie Implementation des OBEX Protokolls für Linux.

Open Source
Software, deren Quellcode frei verfügbar ist.
Siehe auch http://www.gnu.org/philosophy/free-sw.html

OBEX
OBEX steht für OBject Exchange und ist ein Standard, der Datenaustausch über einen IrDA oder Bluetooth Link spezififiziert.

Serielles Kabel
Ein Kabel, das Daten seriell überträgt. Heute im Zusammenhang mit Mobiltelephonen üblich sind „USB zu Mobiltelephon" Verbindungen. Ein Kabel, das PC seitig einen klassischen RS-232 Anschluss aufweist, ist fast nicht mehr zu finden.

SM
Short Message
Dieser Begriff ist in der realen Welt inexistent, deshalb wird eine SM auch in diesem Dokument, obwohl eigentlich falsch, als SMS bezeichnet.

SMS
Short Message Service

11.2 Quellenverzeichnis

11.2.1 Bücher

C++-Entwicklung mit Linux

Author: Thomas Wieland
Erschienen im dpunkt Verlag, Heidelberg, 2002, ISBN 3-932588-74-6

11.2.2 Links

Allgemein

KDevelop – Home
http://kdevelop.kde.org/

OpenOBEX : OpenOBEX
http://openobex.sourceforge.net/

Project `ezV24'-Library
http://ezv24.sourceforge.net/

BlueZ - Official Linux Bluetooth protocol stack
http://www.bluez.org/

Doxygen
http://www.stack.nl/%7Edimitri/doxygen/

OpenOBEX : Applications
http://triq.net/obex/applications.html

pxh: gsmlib
http://www.pxh.de/fs/gsmlib/

Eckhardt Holz Startseite
http://www.informatik.hu-berlin.de/%7Eholz/Vorlesung/HS/

TuxMobil: Debian Packages for (Cell) Phones
http://tuxmobil.org/debian_linux_phones.html

SCMxx
http://www.hendrik-sattler.de/scmxx/

The SonyEricsson P900 and Linux
http://home.tiscali.be/philipa/p900.html

Holtmann Organization
http://www.holtmann.org/

Spezifische Dokumentation

KDE Developer's Corner – Home
http://developer.kde.org/

The KDE API Reference (The KDE API Reference)
http://developer.kde.org/documentation/library/3.2-api/

Qt Reference Documentation (Free Edition)
http://doc.trolltech.com/3.3/index.html

Qt Designer Manual
http://doc.trolltech.com/3.2/designer-manual.html

C++ FAQ LITE
http://www.parashift.com/c%2B%2B-faq-lite/index.html

Doxygen Commands
http://www.stack.nl/%7Edimitri/doxygen/commands.html

The ezV24 Library API
http://ezv24.sourceforge.net/api-html/

AT Test Commands
http://www.anotherurl.com/library/at_test.htm

Sony Ericsson Developers Page
http://renegade.w3xs.com/dev.htm

Exkurs: Versenden von Kurznachrichten mittels PDU
http://www.nobbi.com/sms_pdu.htm

GSM-spezifische AT-Kommandos
http://www.nobbi.com/atgsm.htm

ASCII Table
http://www.neurophys.wisc.edu/www/comp/docs/ascii.html

[Chaostal] Bluetooth for fun and profit
http://www.chaostal.de/cgi-bin/parser.cgi?input=article/bluetooth

Tutorials

KDevelop - 3.0.x tutorials
http://kdevelop.org/index.html?filename=3.0/tutorials.html

Qt Designer and KDevelop-3.0 for beginners
http://women.kde.org/articles/tutorials/kdevelop3/

QT Designer - Quick Start
http://doc.trolltech.com/3.3/designer-manual-2.html

QT Designer - Creating a Main Window Application
http://doc.trolltech.com/3.3/designer-manual-3.html

Einführung in den Qt-Designer 3.0
http://www.lcs-chemie.de/qtdesigner.htm

KDE Tutorial
http://devel-home.kde.org/%7Elarrosa/tutorial/index.html

How to Learn Qt
http://doc.trolltech.com/3.3/how-to-learn-qt.html

Execve
http://www.unet.univie.ac.at/aix/libs/basetrf1/exec.htm

11.3 Sitzungsprotokolle

11.3.1 Sitzungsprotokoll vom 23. März 2004

Technikerschule Uster **TSU**

Protokoll Betreuungssitzung Nr: __1___

Art der Arbeit	O Vordiplomarbeit ⊗ Diplomarbeit O Abschlussarbeit NDS
Student	Thomas Gemperli
Betreuer	Kurt Burri
Thema	„Blue Phone" Linux tool or managing a Sony Ericsson mobile phone
Ort, Datum, Zeit	23.3.04 / 18[so]

Arbeitsstand	Präzision der Aufgabenstellung - Einarbeiten und Analyse Bluetooth-Protokoll Sony Ericsson (Informationsbeschaffung) - Einarbeitung in Entwicklungsumgebung
Probleme, Fragen	KDevelop 3.0 unter LINUX - Analyse und Design der C++ Applikation (inkl. graph. Oberfläche) - Implementation
Weiteres Vorgehen	- Muss-Anforderungen gemäss Vorschlag
Beschlüsse	Aufgaben bis nächste Besprechung - Grundlagen Bluetooth-Protokoll - Planung - Pflichtenheft
Nächster Termin	Termin per Mail vereinbaren (KW 15. Mai)

Der Betreuer gibt dem Studenten jeweils eine Kopie. Er sammelt die ausgefüllten Protokolle und bewahrt sie auf bis 1 Monat nach der Präsentation (Ablauf Rekursfrist). 2.5.2001, HM

Unterschrift Betreuer Unterschrift Student

Freigabe Schulleitung / 2.5.2000 T43405-10 1 von 1
http://www.iso.gotdns.com/tomas/pdf/protokoll betreuungssitzung.doc

11.3.2 Sitzungsprotokoll vom 11. Mai 2004

Technikerschule Uster **TSU**

Protokoll Betreuungssitzung Nr: __2___

Art der Arbeit	O Vordiplomarbeit ☒ Diplomarbeit O Abschlussarbeit NDS
Student	Thomas Gemperli
Betreuer	Kurt Burri
Thema	„Blue Phone" Linux tool or managing a Sony Ericsson mobile phone
Ort, Datum, Zeit	Pfäffikon, 11.5.04 / 18⁰⁰
Arbeitsstand	- Testprogramm mit Bluetooth läuft "OpenObex" ⇒ ohne grafische Oberfläche - Kompatibel zu T68, T610 - BlueZ OpenSource Projekt für Bluetooth - API OpenObex noch nicht bekannt
Probleme, Fragen	- Angespannte Terminsituation heute
Weiteres Vorgehen	- Ferien 3 Wochen vom 10. Juli - Dokugerüst bis anfangs Juli - Vorabklärungen (KDE, Testprogramm) bis anfangs Juli ⇒ Ziel: 100% Nutzung der Ferien!
Beschlüsse	- Weiterentwicklung Testprogramm - Pflichtenheft bis nächste Besprechung
Nächster Termin	11. oder 12. Juni gemäss Mail

Der Betreuer gibt dem Studenten jeweils eine Kopie. Er sammelt die ausgefüllten Protokolle und bewahrt sie auf bis 1 Monat nach der Präsentation (Ablauf Rekursfrist). 2.5.2001, HM

Unterschrift Betreuer Unterschrift Student

Freigabe: Schulleitung / 2.5.2000 T40405.10 1 von 1
http://www.iso.gotons.com/domasy/pdf/protokoll betreuungssitzung.doc

11.3.3 Sitzungsprotokoll vom 30. Juli 2004

Technikerschule Uster **TSU**

Protokoll Betreuungssitzung Nr: __3___

Art der Arbeit	O Vordiplomarbeit ☒ Diplomarbeit O Abschlussarbeit NDS
Student	Thomas Gemperli
Betreuer	Kurt Burri
Thema	„Blue Phone" Linux tool or managing a Sony Ericsson mobile phone
Ort, Datum, Zeit	Pfäffikon, 30.7.04 / 18⁰⁰

Arbeitsstand	• Terminsituation äusserst kritisch! • Ferien konnten nicht wie geplant bezogen werden. (wichtige Server-Migration im Geschäft)
Probleme, Fragen	• Technisch noch nicht alles klar ⇒ Weitere Tools sind notwendig.
Weiteres Vorgehen	• Technische Abklärungen bis So. 1.8. ⇒ Entscheid an K. Burri • Ev. Redimensionierung der Aufgabenstellung schriftlich an Abteilungsvorstand • Gespräch mit Chef
Beschlüsse	• Mail bis So. 1.8. abends Redimensionierung Aufgabenstellung Ja/Nein / Was?
Nächster Termin	

Der Betreuer gibt dem Studenten jeweils eine Kopie. Er sammelt die ausgefüllten Protokolle und bewahrt sie auf bis 1 Monat nach der Präsentation (Ablauf Rekursfrist). 2.5.2001, HM

Unterschrift Betreuer Unterschrift Student

Freigabe: Schulleitung / 2.5.2000 T40405-10 1 von 1
http://www.isc.gotdna.com/domasys/pdf/protokoll betreuungssitzung.doc

11.3.4 Sitzungsprotokoll vom 13. August 2004

Technikerschule Uster **TSU**

Protokoll Betreuungssitzung Nr: __4___

Art der Arbeit	O Vordiplomarbeit ☒ Diplomarbeit O Abschlussarbeit NDS
Student	Thomas Gemperli
Betreuer	Kurt Burri
Thema	„Blue Phone" Linux tool or managing a Sony Ericsson mobile phone
Ort, Datum, Zeit	Pfäffikon, 13.8.04 / 17⁰⁰
Arbeitsstand	• Technisch soweit abgeschlossen ⇒ Bilder & Töne herunterladen nicht impl. ⇒ Kalender sichern nicht impl.
Probleme, Fragen	
Weiteres Vorgehen	• Aufstellen der Applikation: - Mit Shell oder in Applikation einbauen • Dokumentation fertig stellen • Applikation statisch linken • Debian-Package / RPM
Beschlüsse	—
Nächster Termin	—

Der Betreuer gibt dem Studenten jeweils eine Kopie. Er sammelt die ausgefüllten Protokolle und bewahrt sie auf bis 1 Monat nach der Präsentation (Ablauf Rekursfrist). 2.5.2001, HM

Unterschrift Betreuer Unterschrift Student

Freigabe: Schulleitung / 2.5.2000 T40405-10 1 von 1

http://www.iso.gotdns.com/domasy/pdf/protokoll betreuungssitzung.doc

11.4 CD

11.4.1 Inhaltsverzeichnis der CD

Verzeichnis	Inhalte
documentation	• Diese Dokumentation im PDF Format • Die Benutzeranleitung • Die Doxygen generierte Klassen-Dokumentation der Software
source	• Der Quellcode von BluePhone, bereit zum kompilieren
packages	• Die Installationspackete der Software im tar.gz, dpkg und rpm Format